НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ
(повна назва інституту/факультету)

КОНСТРУЮВАННЯ ЕЛЕКТРОННО-ОБЧИСЛЮВАЛЬНОЇ АПАРАТУРИ
(повна назва кафедри)

«На правах рукопису»
УДК: 003.26;
004.056.55

«До захисту допущено»

Завідувач кафедри КЕОА

_____ О.М.Лисенко
(підпис) (ініціали, прізвище)

“_18”__травня__2020 р.

Магістерська дисертація

зі спеціальності (спеціалізації) 172 – Телекомунікації та радіотехніка
(код і назва спеціальності)

на тему: Метод узгодження ключа на основі LDPC-протоколу в системах квантового розподілу ключів

Виконав: студент _6_ курсу, групи ДК-81мн
(шифр групи)

Білаш Богдан Олегович _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник зав. кафедри КЕОА, д.т.н., проф. Лисенко О.М. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент доц. кафедри АМЕС, к.т.н., доц. Швайченко В.Б. _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут
імені Ігоря Сікорського»

Інститут/факультет _____електроніки_____
(повна назва)

Кафедра_____конструювання електронно-обчислювальної апаратури_____
(повна назва)

Рівень вищої освіти – другий (магістерський) за освітньо-науковою програмою

Спеціальність (спеціалізація) _172 – Телекомунікації та радіотехніка_____
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Лисенко О.М._
(підпис) (ініціали, прізвище)
«04»_лютого__2020 р.

ЗАВДАННЯ

на магістерську дисертацію
студенту Білашу Богдану Олеговичу__
(прізвище, ім'я, по батькові)

1. Тема дисертації Метод узгодження ключа на основі LDPC-протоколу в системах квантового розповсюдження ключів

науковий керівник дисертації Лисенко Олександр Миколайович, д.т.н., проф.,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «_17_»_березня_ 2020р. № 887-с

2. Строк подання студентом дисертації _____15.05.2020р._____

3. Об'єкт дослідження _процес передачі даних в квантовій системі QKD_

4. Предмет дослідження методи узгодження ключів та алгоритми їх реалізації в системах QKD

5. Перелік завдань, які потрібно розробити 1. Аналіз існуючих класичних та квантових криптографічних алгоритмів і методів корекції помилок. 2. Обгрунтування вибору базового методу Валенти виправлення помилок у системах QKD та його моделювання. 3. Модифікований метод узгодження ключа на основі LDPC кодів в системах QKD та його дослідження. 4. Розроблення стартап-проекту

6. Перелік графічного (ілюстративного) матеріалу

Презентація у форматі PowerPoint

7. Орієнтовний перелік публікацій 1 публікація

8. Консультанти розділів дисертації*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

9. Дата видачі завдання 04.02.2020р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів магістерської дисертації	Примітка
1	Аналіз тематичної літератури	04.02.20—15.02.20	
2	Обґрунтування вибору методу узгодження ключів в системах QKD, визначення шляхів вдосконалення методу	16.02.20—29.02.20	
3	Модифікація методу, розроблення алгоритмічних рішень його реалізації	01.03.20—15.03.20	
4	Програмна реалізація методу та моделювання обраного рішення	16.03.20—20.04.20	
5	Розробка стартап-проекту	21.04.20—30.04.20	
6	Оформлення дисертації	01.05.20—18.05.20	

Студент _____ Білаш Б.О.
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ Лисенко О. М.
(підпис) (ініціали, прізвище)

РЕФЕРАТ

Магістерська дисертація складається з 129 сторінок, в якій міститься 21 рисунок, 22 таблиці, використано 42 джерела.

Актуальність. Відомо, що наразі область квантової криптографії інтенсивно розвивається в системах квантового розповсюдження ключів (quantum key distribution, QKD). Системи QKD мають як переваги, зокрема, створення справді випадкового ключа, забезпеченого законами квантової фізики, так і недоліки, наприклад, помилки, які виникають із-за шумів у ненадійному квантовому каналі та аномалій, викликаних третьою стороною, які знижають його надійність. Наразі розвиваються два напрямки підвищення надійності систем QKD: дослідження квантового каналу на фізичному рівні та дослідження виправлення помилок після створення просіяного ключа. Саме тому розробка нових та удосконалення існуючих методів і засобів виявлення та усунення помилок в системах QKD наразі є важливим та **актуальним** завданням.

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційні дослідження проводилися в Центрі квантової інформації (*Center for Quantum Information*) Корейського інституту науки і технологій (*Korea Institute of Science and Technology, KIST*) відповідно до тематики наукових досліджень цього закладу, наукових напрямків діяльності кафедри КЕОА, а також пріоритетного напрямку розвитку науки і техніки України "Інформаційні та комунікаційні технології".

Метою дисертаційної роботи є підвищення надійності та захищеності систем QKD шляхом подальшого розвитку методу узгодження ключа із застосуванням корекції помилок на основі LDPC-кодів та розроблення алгоритмічних і програмних рішень його реалізації.

Для досягнення поставленої мети в роботі вирішувалися наступні **задачі:**

- проаналізовано існуючі класичні та квантові криптографічні алгоритми і методи корекції помилок;

- обґрунтовано вибір в якості базового для подальшого удосконалення метод Валенти виправлення помилок у системах QKD та проведено його моделювання;
- модифіковано метод узгодження ключа на основі LDPC-кодів в системах QKD та виконано його дослідження;
- виконано розроблення стартап-проекту.

Об'єктом дослідження є процес передачі даних в квантовій системі QKD.

Предметом дослідження є методи узгодження ключів та алгоритми їх реалізації в системах QKD**.**

Методами дослідження є Cascade, Winnow, LDPC, метод Валенти, метод Міліцевича, моделювання обраних методів.

Наукова новизна отриманих результатів дослідження полягає в наступному:
- удосконалено метод узгодження ключа в системах QKD шляхом виявлення та корекції помилок на основі LDPC-кодів, який відрізняється від відомого методу Валенти введенням процедури виправлення помилкових повідомлень за рахунок перебору можливих варіантів нових повідомлень та порівняння їх синдромів з синдромом повідомлення передавальної сторони, що дозволило підвищити надійність систем QKD;
- запропоновано використання LPDC *H*-матриці перевірки, яка не буде створювати однакові синдроми повідомлень до трьох помилок, що дало змогу підвищити надійність систем QKD.

Практичне значення отриманих результатів визначається створеними алгоритмічними та програмними рішеннями реалізації запропонованого модифікованого методу узгодження ключа в QKD системах. Створено програмний комплекс на мові програмування C та проведено моделювання результатів в програмі GNU Octave. Планується впровадження одержаних результатів в Центрі квантової інформації (*Center*

for Quantum Information) Корейського інституту науки та технологій (*Korea Institute of Science and Technology, KIST*).

Апробація результатів дисертації. Результати дисертаційних досліджень апробовано на X Міжнародній науково-практичній інтернет-конференції «Сучасний рух науки», м. Дніпро, квітень, 2020р.

Публікації. За матеріалами дисертації опубліковано 1 друковану працю в збірнику матеріалів конференції (див. Додаток А):

- Bilash Bohdan. The Implementation of the Modified Error Correction Method in Quantum Key Distribution // Збірник тез доповідей X Міжнародної науково-практичної інтернет-конференції «Сучасний рух науки», м. Дніпро, квітень, 2020 р.– С. 109-112.

За матеріалами досліджень також підготовлено та подано до друку у фаховому виданні України 1 статтю (реєстр. № 201253, див. Додаток Б), яка наразі проходить процедуру наукового рецензування:

- Bilash B.O. Optimal low density parity check matrices to correct quantum key errors for QKD // Мікросистеми, Електроніка та Акустика. – 2020.

Ключові слова: система QKD, LDPC, корекція помилок, узгодження ключа, синдром, повідомлення, метод, Валента, алгоритм.

ABSTRACT

The master's dissertation consists of 129 pages, which contains 21 figures, 22 tables, used 42 sources.

The relevant. It is known that the field of quantum cryptography is currently developing intensively in quantum key distribution (QKD) systems. QKD systems have both advantages, such as creating a truly random key provided by the laws of quantum physics, and disadvantages, such as errors due to noise in an unreliable quantum channel and anomalies caused by a third party that reduce its reliability. Currently, two directions are being developed to increase the reliability of QKD systems: the study of the quantum channel at the physical level and the study of error correction after the creation of the sieved key. That is why the development of new and improvement of existing methods and tools for detecting and eliminating errors in QKD systems is currently an important and **relevant** task.

Relationship of work with scientific programs, plans, themes. Dissertation research was conducted at the *Center for Quantum Information* of the *Korea Institute of Science and Technology (KIST)* in accordance with the research topics of this institution, scientific activities of the Department of ECED, as well as the priority of science and technology. Of Ukraine “Information and communication technologies”.

The purpose of the dissertation is to increase the reliability and security of QKD systems by further developing the method of key matching using error correction based on LDPC-codes and the development of algorithmic and software solutions for its implementation.

To achieve this goal, the following **tasks** were solved in the work:

• the existing classical and quantum cryptographic algorithms and methods of error correction are analyzed;

• the choice of Valenta's method of error correction in QKD systems as a basic one for further improvement is substantiated and its modeling is carried out;

• the method of key matching based on LDPC codes in QKD systems was modified and its research was performed;

• development of a startup project was performed.

The object of research is the process of data transmission in the quantum system QKD.

The subject of research is the methods of key matching and algorithms for their implementation in QKD systems.

Research **methods** are Cascade, Winnow, LDPC, Valenta's method, Militsevich's method, modeling of selected methods.

The scientific novelty of the results of the study is as follows:
• Improved the method of key matching in QKD systems by detecting and correcting errors based on LDPC-codes, which differs from the known method of Valenta by introducing a procedure for correcting erroneous messages by searching for possible variants of new messages and comparing their syndromes with the transmitter's message syndrome. reliability of QKD systems;
• It is proposed to use the LPDC H-matrix of the test, which will not create the same message syndromes up to three errors, which allowed to increase the reliability of QKD systems.

The practical significance of the obtained results is determined by the created algorithmic and software solutions for the implementation of the proposed modified method of key matching in QKD systems. A software package in the C programming language was created and the results were modeled in the GNU Octave program. It is planned to implement the obtained results in the *Center for Quantum Information* of the *Korea Institute of Science and Technology (KIST)*.

Approbation of the results of the dissertation. The results of dissertation research were tested at 10th International Scientific and Practical Internet Conference “Modern Movement of Science”, Dnipro, April 2020.

Publications. Based on the materials of the dissertation, 1 paper was published in the collection of conference materials:

- Bilash Bohdan. The Implementation of the Modified Error Correction Method in Quantum Key Distribution //Collection of articles of the 10th International Scientific and Practical Internet Conference “Modern Movement of Science”, Dnipro, April 2020. - P.P. 109-112.

According to the research materials, 1 article was also prepared and submitted for publication in the professional publication of Ukraine (Reg. № 201253), which is currently undergoing a scientific review procedure:

- Bilash B.O. Optimal low density parity check matrices to correct quantum key errors for QKD //Microsystems, Electronics and Acoustics – 2020.

Key words: QKD, LDPC, error correction, key reconciliation, message, method, Walenta, algorithm.

Зміст

Перелік умовних позначень

Відкритий текст – у криптографії вихідний текст, що підлягає шифруванню або такий, що вийшов у результаті розшифрування шифртексту; може бути прочитаний без додаткової обробки (без розшифрування).

Виявлення та виправлення помилок – контроль цілісності даних при записі і відтворенні інформації або при їх передачі по лініях зв'язку, а також забезпечення відновлення інформації після читання її з пристрою зберігання або каналу зв'язку; для виявлення помилок використовують коди виявлення помилок, для виправлення - коригувальні коди (коди, що виправляють помилки, коди з корекцією помилок, завадостійкі коди).

Квант – у фізиці частка певної величини, яка не ділиться; це загальна назва певних порцій енергії, моменту кількості руху та інших величин.

Ключ – секретна інформація, яка використовується криптографічним алгоритмом при зашифровуванні / розшифровуванні повідомлень; при використанні одного і того ж алгоритму результат шифрування залежить від ключа; в сучасній криптографії втрата ключа призводить до практичної неможливості розшифрувати повідомлення.

Кодування – фіксоване перетворення інформації з одного виду в інший; у кодуванні відсутнє поняття ключа.

Криптологія – розділ науки, що вивчає методи шифрування і дешифрування інформації.

Криптографія – розділ криптології про методи забезпечення конфіденційності (неможливості прочитання інформації сторонніми особами), цілісності даних (неможливості непомітної зміни інформації), аутентифікації (перевірки справжності автора чи інших властивостей об’єкта).

Криптоаналіз – розділ криптології, що займається математичними методами порушення конфіденційності і цілісності інформації без знання ключа.

Криптографічна стійкість – здатність криптографічного алгоритму протистояти криптоаналізу; стійким вважається алгоритм, який для успішної атаки вимагає від противника недосяжних обчислювальних ресурсів, недосяжного обсягу перехоплених відкритих і зашифрованих повідомлень чи ж такого часу розкриття, який по його закінченню захищена інформація буде вже не актуальною.

Обчислювальна складність – це поняття теорії складності обчислень для оцінки ресурсів, необхідних для виконання алгоритму, за відповідними критеріями, зокрема, часу для розв'язання задачі та обсягу необхідної пам'яті при збільшенні розміру вхідних даних.

Шифртекст – термін криптології, яким називають результат шифрування відкритого тексту.

Шифрування – оборотне перетворення інформації з метою приховування інформації від неавторизованих осіб з наданням в цей же час авторизованим користувачам доступу до неї; важливою особливістю будь-якого алгоритму шифрування є використання ключа, який

визначає вибір конкретного перетворення із сукупності можливих для даного алгоритму.

BEC	– Backward Error Correction
FEC	– Forward Error Correction
LDPC	– Low Density Parity Check
QKD	– Quantum Key Distribution
QBER	– Quantum Bit Error Rate

Вступ

Актуальність. Необхідність захисту інформації супроводжувала людство на протязі всієї історії. Це мало на увазі захист як стратегічно важливих документів, так і прагнення захистити свою приватну інформацію. Першим найбільш відомим методом шифрувати інформацію був шифр Цезаря [1]. В такому шифрі ключем виступає сам метод шифрування. Поступово людство вдосконалювало методи шифрування, де ключами вже виступали дійсно випадкові або математично створенні за певним алгоритмом дані. Найбільш відомим алгоритмом другого прикладу є RSA (по першим буквам прізвищ вчених) [2], де існує відкритий ключ, який може бути перехоплений будь-ким, але через високу складність декодування дуже тяжко відтворити вихідний текст, якщо невідомо секретний ключ. Даний шифр є прикладом класичного криптографічного шифру, який базується на математичному апараті.

Як альтернатива згаданим вище криптографічним шифрам, завдяки успішному вивченню та розвитку квантової фізики в останні роки бурливо розвивається квантова криптографія, яка ґрунтується на тому, що секретний ключ створюється на законах квантової фізики [3].

Відомо, що наразі область квантової криптографії інтенсивно розвивається в системах квантового розповсюдження ключів (quantum key distribution, QKD). Системи QKD мають як переваги, зокрема, створення справді випадкового ключа, забезпеченого законами квантової фізики, так і недоліки, наприклад, помилки, які виникають із-за шумів у ненадійному квантовому каналі та аномалій, викликаних третьою стороною, які знижають його надійність. Наразі розвиваються два напрямки підвищення надійності систем QKD: дослідження квантового каналу на фізичному рівні та дослідження виправлення помилок після створення просіяного ключа. Існує багато методів корекції помилок, які застосовуються саме в системах QKD. Однак, серед них найбільш широкого застосування знайшли методи Cascade,

Winnow та LDPC (Low Density Parity Check). При цьому, деякі методи створені спеціально для систем QKD, інші - адаптовані з класичних методів корекції помилок. Кожний з методів має свої як сильні, так і слабкі сторони, які необхідно враховувати. Саме тому розробка нових та удосконалення існуючих методів і засобів виявлення та усунення помилок в системах QKD наразі є важливим та **актуальним** завданням.

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційні дослідження проводилися в Центрі квантової інформації (*Center for Quantum Information*) Корейського інституту науки і технологій (*Korea Institute of Science and Technology, KIST*) відповідно до тематики наукових досліджень цього закладу, наукових напрямків діяльності кафедри КЕОА, а також пріоритетного напрямку розвитку науки і техніки України "Інформаційні та комунікаційні технології".

Метою дисертаційної роботи є підвищення надійності та захищеності систем QKD шляхом подальшого розвитку методу узгодження ключа із застосуванням корекції помилок на основі LDPC-кодів та розроблення алгоритмічних і програмних рішень його реалізації.

Для досягнення поставленої мети в роботі вирішувалися наступні **задачі:**

- проаналізовано існуючі класичні та квантові криптографічні алгоритми і методи корекції помилок;
- обґрунтовано вибір в якості базового для подальшого удосконалення метод Валенти виправлення помилок у системах QKD та проведено його моделювання;
- модифіковано метод узгодження ключа на основі LDPC-кодів в системах QKD та виконано його дослідження;
- виконано розроблення стартап-проекту.

Об'єктом дослідження є процес передачі даних в квантовій системі QKD.

Предметом дослідження є методи узгодження ключів та алгоритми їх реалізації в системах QKD**.**

Методами дослідження є Cascade, Winnow, LDPC, метод Валенти, метод Міліцевича, моделювання обраних методів.

Наукова новизна отриманих результатів дослідження полягає в наступному:

- удосконалено метод узгодження ключа в системах QKD шляхом виявлення та корекції помилок на основі LDPC-кодів, який відрізняється від відомого методу Валенти введенням процедури виправлення помилкових повідомлень за рахунок перебору можливих варіантів нових повідомлень та порівняння їх синдромів з синдромом повідомлення передавальної сторони, що дозволило підвищити надійність систем QKD;
- запропоновано використання LPDC *H*-матриці перевірки, яка не буде створювати однакові синдроми повідомлень до трьох помилок, що дало змогу підвищити надійність систем QKD.

Практичне значення отриманих результатів визначається створеними алгоритмічними та програмними рішеннями реалізації запропонованого модифікованого методу узгодження ключа в QKD системах. Створено програмний комплекс на мові програмування C та проведено моделювання результатів в програмі GNU Octave. Планується впровадження одержаних результатів в Центрі квантової інформації (*Center for Quantum Information*) Корейського інституту науки та технологій (*Korea Institute of Science and Technology, KIST*).

Апробація результатів дисертації. Результати дисертаційних досліджень апробовано на X Міжнародній науково-практичній інтернет-конференції «Сучасний рух науки», м. Дніпро, квітень, 2020р.

Публікації. За матеріалами дисертації опубліковано 1 друковану працю в збірнику матеріалів конференції (див. Додаток А):

- Bilash Bohdan. The Implementation of the Modified Error Correction Method in Quantum Key Ditribution // Збірник тез доповідей X Міжнародної науково-практичної інтернет-конференції «Сучасний рух науки», м. Дніпро, квітень, 2020 р.– С. 109-112.

За матеріалами досліджень також підготовлено та подано до друку у фаховому виданні України 1 статтю (реєстр. № 201253, див. Додаток Б), яка наразі проходить процедуру наукового рецензування:

- Bilash B.O. Optimal low density parity check matrices to correct quantum key errors for QKD // Мікросистеми, Електроніка та Акустика. – 2020.

Структура дисертаційної роботи містить вступ, 4 розділи, загальний висновок, перелік використаної літератури та додатки:

- Додаток А. Копія тез доповіді на конференції;
- Додаток Б. Копія публікації у фаховому виданні;
- Додаток В. Відгук про роботу;
- Додаток Г. Лістинг програми.

РОЗДІЛ 1. АНАЛІЗ КЛАСИЧНИХ ТА КВАНТОВИХ КРИПТОГРАФІЧНИХ АЛГОРИТМІВ І МЕТОДІВ КОРЕКЦІЇ ПОМИЛОК

Як добре відомо, криптографія і шифрування вже тисячі років використовуються людьми для захисту своїх секретів. Елементи криптографії вже спостерігались в Древніх Єгипті та Греції, а найбільш ранній відомий спосіб військового застосування криптографії належить Гай Юлію Цезарю. Близько 2000 років тому Цезар, будучі полководцем римської армії, вирішив проблему безпечних комунікацій зі своїми полками, яка полягала в тому, що гінці з секретними військовими повідомленнями часто перехоплювалися ворогом. Цезар розробив шифр підстановки, в якому заміняв одні букви іншими, що були зсунуті на певну кількість вперед (циклічно). Наприклад, якщо український алфавіт зсунути на дві літери вперед, то літери А Б В Г... відповідно заміняться на В Г Ґ Д... і тд. Але у випадку українського алфавіту, ми можемо легко знайти ключ розшифрування даного шифру, адже ми маємо всього 32 можливі варіанти зсунення відповідної букви (в українському алфавіті 33 букві, відповідно букву А ми можемо замінити 32 іншими буквами).

Наступним кроком розвитку шифру Цезаря став шифр простої заміни (рис. 1.1), який на відміну від шифру Цезаря використовує спеціальну таблицю підстановки, згідно якій кожна літера може бути замінена іншою літерою в довільному порядку. З цього витікає, що першу літеру українського алфавіту можна замінити 32 іншими літерами, другу - вже 31 способом, третю - 30 і т. д. Цей приклад є умовно простим, адже можливі ситуації, коли дві букви взаємно замінюють одна іншу, але, зазвичай, маємо 32! різних варіантів створення таблиці підстановки. Тільки той, хто знав цю таблицю, міг розшифрувати секретне повідомлення. Отже, шифр простої заміни виглядає більш надійнішим, аніж шифр Цезаря [4].

Але і цей шифр через деякий час був «зламаний» завдяки використанню частотного аналіза, який передбачає, що частота появи заданої букви алфавіту

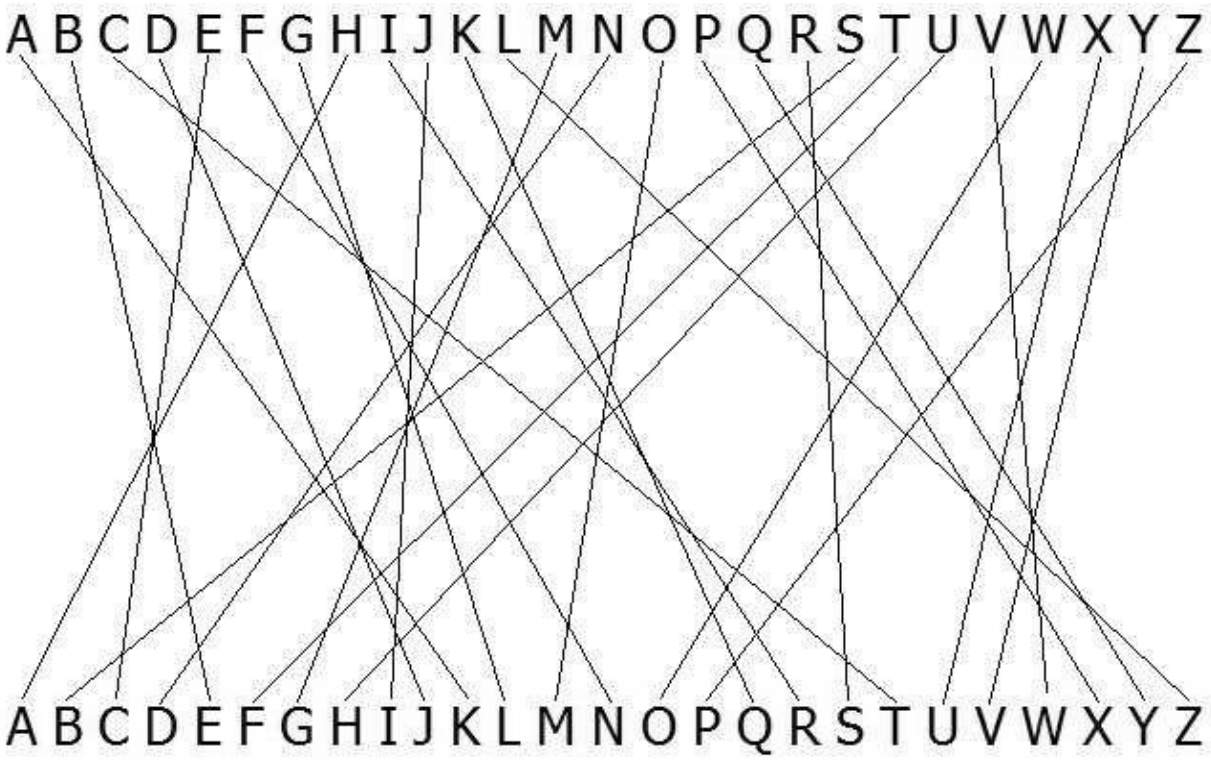


Рисунок 1.1 - Приклад шифру простої заміни

в достатньо довгих текстах є однією і тою ж для різних текстів одної мови. Тобто, довжина тексту має бути достатньо великою. Якщо порахувати частоту появи одної букви в тексті і розділити на загальну кількість букв, знайдемо частоту, з якою зустрічається ця буква. На даний момент частотність букв будь-якої мови вже є достатньо точно порахованою і знайти її можна в мережі Інтернет.

Також можливо використовувати смисловий підхід. Наприклад можливе часте згадування імені адресата в тексті, тобто певний порядок літер буде зустрічатись доволі часто.

Або спостережливість, що в українській мові літера “і” виступає сполучником між словами. Тобто, якщо замінити її на літеру “м”, ми будемо часто бачити просто літеру “м” в тексті, що дає можливість передбачати, що дана літера замінює літеру “і”, а далі методом заміни замінювати всі літери “м” на літери “і” та пробувати знаходити інші логічні спостереження.

Поступово, шифри змінювались, покращувались, зламати їх ставало тяжче, але було можливо. Адже яким б складним шифр не був — він мав певну закономірність, завдяки який даний шифр вдавалось зламати. Останнім відомим прикладом можна згадати роторну шифрувальну машину “Enigma”, яка використовувалась у часи Другої світової війни [5]. Ця машина дозволяла швидко шифрувати та розшифрувати повідомлення а її алгоритм роботи був відомий. Але підібрати ключ було досить тяжко через те, що кількість можливих комбінацій була надзвичайно великою, а сучасних комп’ютерних ресурсів для того, щоб перебрати усі можливі варіанти ключа, тоді не було. Але і дану машину через певний час вдалось зламати.

Можна зробити висновок, що такі шифри, ключі яких базуються на підстановах одних символів на інші не виявились достатньо криптографічно стійкими. Пізніше почали з’являтись шифри, алгоритми шифрування та ключі яких вже базувались на математичних властивостях та законах.

1.1. Види шифрування

Основним компонентом криптографії є шифрування. Повідомлення шифруються і розшифровуються за допомогою складних алгоритмів, створених комбінацією інформатики та математики. Шифрування використовує алгоритм і ключ для перетворення вхідних даних в зашифровані вихідні дані. Цей метод захисту дозволяє переглядати повідомлення виключно відправнику і одержувачу, оскільки зашифровану інформацію може прочитати тільки той, хто має секретний ключ для перетворення повідомлення в простий текст. В сучасній криптографії існує два типи шифрів в залежності від застосування кількості ключів: симетричний та асиметричний шифр.

Асиметричний шифр. Цей шифр також називають шифром з відкритим ключем криптографії (public-key cryptography, PKC). Алгоритм PKC

використовує два ключа: відкритий і закритий. Відкритий може бути відомий багатьом та передається по відкритому каналу. Розшифрувати дані з його допомогою неможливо. Наприклад, адреса електронної пошти є відкритим ключем. Закритий ключ є секретним, використовується для розшифровки повідомлення, ніколи не розкривається іншій стороні. Наприклад, пароль облікового запису електронної пошти є ключем до відкриття електронних листів. Не має значення, який ключ застосовується в першу чергу, але для роботи необхідні обидва. Дані можуть бути зашифровані за допомогою відкритого або закритого ключа.

Симетричний шифр. Є найпростішим алгоритмом, який з’явився раніше, ніж асиметричний шифр. Криптографи часто називають його секретним ключем криптографії (secret-key cryptography, SKC) або загальним, оскільки шифрування і розшифрування інформації відбувається з використанням одного і того ж ключа. Симетричне шифрування має на увазі, що секретний ключ повинен бути відомий як одержувачу, так і відправнику.

В залежності від необхідності шифрувати весь текст одразу чи по мірі його надходження розділяють блочний та потоковий шифри. Обидва шифри є різновидами блочного шифру.

Блочний шифр. Мається на увазі, що кожен блок даних шифрується або розшифровується окремо, причому кожен біт в вихідному блоці залежить від кожного біта в відповідному вхідному блоці, але не від інших бітів цього блока. Розмір блоку визначається алгоритмом. У більшості випадків блоки зазвичай мають 64-або 128-розрядний формат. Це означає, що їх розмір визначений і залишається завжди незмінним.

Потоковий шифр - це симетричний шифр, в якому кожен символ відкритого тексту перетворюється в символ шифрованого тексту в залежності не тільки від використовуваного ключа, а й від його розташування в потоці відкритого тексту. Потоковий шифр реалізує інший підхід до симетричного шифрування, ніж блокові шифри.

Окремо необхідно виділити хеш-функцію. Хеш-функції є алгоритмами, які в деякому сенсі не використовують ключ. Їх також називають дайджестами повідомлень або одностороннім шифруванням. За допомогою алгоритмів хешування можливо перетворення великих обсягів інформації в рядок двійкових чисел (бітів) певної довжини (хеш), яку важко імітувати. Таким чином хеш-функції забезпечують вимірювання цілісності надісланих файлів. Два різних повідомлення, що містять різну інформацію, не можуть мати однаковий хеш (насправді можуть, але знайти такі повідомлення, які б давали однаковий хеш надзвичайно складно, і вірогідність отримання однакового хешу для різних повідомлень надзвичайно мала). Хеш може використовуватися в якості цифрового підпису або для шифрування і зберігання паролів.

1.2. Шифр Вернама

У 1917 році американський інженер Гільберт Вернам запатентував систему симетричного шифрування, яка отримала назву шифр Вернама або одноразовий блокнот (one-time pad). Д Дана система має дуже просту в побудові структуру: шифртекст створюється через логічну операцію XOR між відкритим текстом та ключем (у випадку, якщо відкритий текст та ключ є двійковими числами). Застосування шифру Вернама дуже схоже на застосування шифру простої заміни. Відмінність заключається в тому, що у шифрі Вернама ключем є не сам метод шифрування, а дійсно випадкова послідовність символів. Для кожного нового відкритого тексту має генеруватися новий ключ однакової довжини як і відкритий текст. Незважаючи на простоту побудови, шифр Вернама володіє абсолютною криптографічною стійкістю, яку у 1945 році довів американський математик Клод Шеннон [6].

Шифр Вернама не знайшов належного застосування у свій час. Однак, наразі він привертає всі більшої уваги у науковому світі. Дослідники

намагаються модернізувати цей шифр сучасними методами криптографії та апаратним забезпеченням. Принцип роботи шифру Вернама можна застосовувати також і в квантовій криптографії [7], [8].

1.3. Сучасна криптографія та шифр RSA

Однією з головних задач сучасної криптографії є вирішення завдання передачі абсолютно секретного ключа по відкритому каналу даних. Однак, розв'язати цю задачу не представляється можливим на даний момент. Тому було запропоновано передачу по відкритому каналу відкритого ключа, який може перехопити кожен.

В 1977 вченими Рональдом Рівестом, Аді Шаміром, Леонардом Адлеманом із MIT запропоновано алгоритм RSA (по першим буквам прізвищ вчених), який є асинхронним алгоритмом шифрування (з відкритим ключем) Алгоритм RSA базується на факторизації простих чисел, однак прості множники можуть мати тисячі знаків. В такому випадку обчислювальній машині необхідна дуже велика кількість часу для повного перебору усіх можливих варіантів.

Також відомими алгоритмами шифрування є:

- Схема Ель-Гамаля (дискретне логарифмування) [9];
- Криптосистема Рабіна (добування кореня) [10];
- Elliptic-curve cryptography (ECC, дискретне логарифмування) [11] і тд.

Наразі алгоритм RSA є одним з популярних алгоритмів шифрування повідомлень, який використовується в повсякденні кожним з нас, навіть не підозрюючи про це. Це пояснюється тим, що в його основі лежить звичайна математика і сам алгоритм створення ключів є дуже простим. Інші представлені вище алгоритми в своїй основі мають дещо складніші математичні процеси. На факторизацію ключа протоколу RSA можуть витратитись тисячі років, але зараз

починають практикуватися методи факторизації чисел за допомогою квантових комп’ютерів (квантовий алгоритм Шора [12]). В 2001 році дослідники з IBM змогли продемонструвати роботу алгоритму, розклавши число 15 на множники 3 і 5 за допомогою квантового комп’ютера з 7 кубітами [13]. Це створює потенційну загрозу алгоритму RSA в майбутньому. Тому дослідники паралельно даному алгоритму шукають інші методи криптографії. Одним із напрямків в цьому стала квантова криптографія, яка базується на законах квантової фізики.

1.4. Основи квантової криптографії

Квантова криптографія ґрунтується на тому, що ключ створюється на законах квантової фізики.

Метод передачі ключа, який використовує квантові явища для гарантії безпечного зв’язку, називають квантовим розповсюдженням ключів (quantum key distribution, QKD). Важливою і унікальною властивістю QKD є можливість виявити присутність третьої сторони, яка намагається отримати інформацію про ключі. Тут використовується фундаментальний аспект квантової механіки: процес вимірювання квантової системи порушує її. Третя сторона, яка намагається отримати ключ, повинна виміряти надіслані квантові стани, що веде до їх зміни і появи аномалії. За допомогою квантової суперпозиції, квантового заплутування передачі даних в квантових станах можна здійснити канал зв'язку, який виявляє аномалії. Якщо кількість аномалій нижче певного порогу, то ключ буде створено, що гарантує безпеку, інакше секретний ключ не буде створено і зв'язок припиняється.

Для QKD виділилися два основних напрямки розвитку систем розподілу ключів.

Перший напрям розвитку оснований на ефекті квантового заплутування [14], [15]. Дві квантово-механічні системи (які можуть бути розділенні

просторово на дуже великі відстані) можуть знаходитись у стані кореляції. Вимірювання обраної величини, яке здійснюється над однією із систем, визначає результат вимірювання цієї величини над другою системою. Стан двох часток зі спіном 1/2 може бути прикладом заплутаного стану. Вимірювання, яке проводиться над однією з двох підсистем, дає результат з однаковою вірогідністю “0” або “1”. Стан другої підсистеми буде протилежним. Квантове заплутування базується на принципі невизначності Гейзенберга. Відомим протоколом квантового розподілу ключа на основі квантового заплутування є протокол EPR (Einstein-Podolsky-Rosen paradox [16]), або E91 [17]. Основними носіями інформації тут, як правило, є електрони.

Інший напрям ґрунтується на кодуванні квантового стану одиничної частки і базується на принципі неможливості розрізнити абсолютно надійно два неортогональні квантові стани [18]. Закони квантової механіки не дозволяють абсолютно надійно розрізняти два квантові стани, якщо вони не є ортогональними. Захищеність першого напряму ґрунтується на теоремі про заборону клонування невідомого квантового стану [19] (яка базується на принципу невизначності Гейзенберга [20]). Неможливо створити точну копію невідомого квантового стану без впливу на початковий стан. Якщо стан квантової частки заданий поляризацією під певним кутом, після прослуховування зловмисником частка матиме інший, змінений під час прослуховування, кут поляризації. В результаті прослуховування квантового каналу приводить до помилок передачі, які можуть бути виявлені легальними користувачами. Першим протоколом квантової криптографії на одночасткових станах став протокол BB84 [21]. Основними носіями інформації тут, як правило, є фотони.

Хоча перший протокол BB84 для реалізації методу QKD був запропонований у 1984 році, цей напрям є новим у сучасній науці та активно досліджується. В якості прикладів науково-дослідних робіт за тематикою QKD можна привести розроблення систем зменшення розмірів мікросхем [22],

комунікація на великих відстанях [23], висока безпечна швидкість передачі ключа QKD [24]–[26], ефективна пост-обробка [27] тощо.

На етапі пост-обробки відбувається узгодження ключа. На другому етапі дві сторони мають просіяні ключі, які відрізняються між собою на величину, рівну QBER. Узгодження ключа відбувається на третьому етапі корекції помилок, де просіяні ключі змінюють таким чином, щоб узгодити їх біти. Після узгодження ключа обидві сторони мають однакові ключі, які поступають на наступний етап. Надалі під узгодженням ключа ми будемо розуміти саме корекцію помилок в просіяних ключах, яка відбувається на третьому етапі.

1.4.1. Протокол BB84

Розглянемо детальніше перший протокол QKD, який називається BB84, створений в 1984 році. В протоколі використовуються 4 квантові стани фотонів. В даному випадку це вектор поляризації фотона. Передавальна сторона (в подальшому - Аліса) обирає в залежності від вектору поляризації фотону значення біта, що передається: 90^0 або 135^0 для “1”, 0^0 або 45^0 для “0”. Одна пара квантових станів належить до базису “+”, інша належить до базису “×”. Всередині обох базисів стани ортогональні, але стани з різних базисів є попарно неортогональними. Базиси повернуті друг відносно друга на 45^0.

Етапи формування ключа:

Аліса випадковим чином обирає один із базисів. Після цього всередині обраного базису випадково обирає один із станів, який відповідає “0” або “1”, після чого посилає фотони.

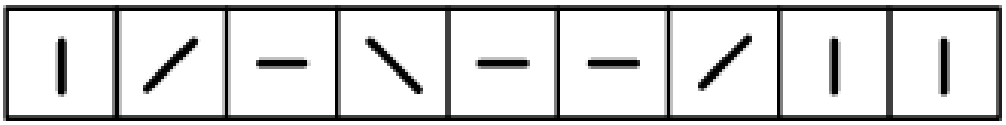


Рисунок 1.2 - Фотони з різною поляризацією

Приймальна сторона (в подальшому - Боб) випадково і незалежно від Аліси обирає для кожного фотона прямолінійний “+” або діагональний “×” базис.

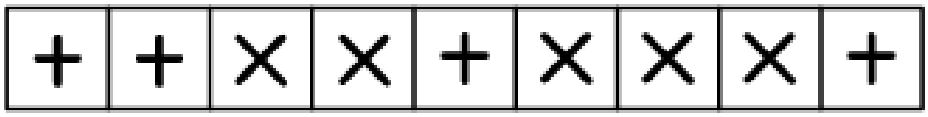


Рисунок 1.3 - Обраний тип вимірювання

Після цього Боб зберігає результати вимірювань.

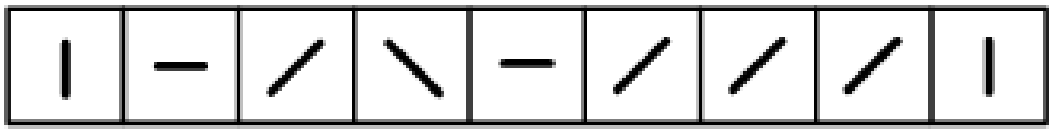


Рисунок 1.4 - Результати вимірювань

Боб по відкритому загальному каналу повідомляє, який тип вимірювання він використав для кожного фотону, тобто який був обраний базис, але результати вимірювань залишаються в секреті.

Аліса повідомляє Бобу по відкритому загальному каналу, які вимірювання були обрані відповідно до початкового базису Аліси.

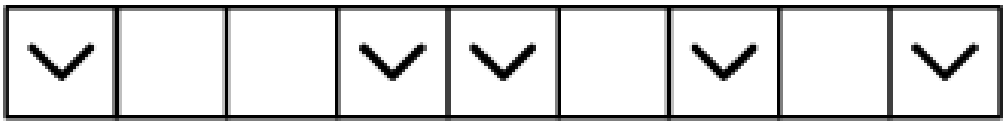


Рисунок 1.5 - Випадки правильних вимірювань

Далі обидва користувачі залишають лише ті випадки, в яких обрані базиси збіглися. Ці випадки переводять в біти (0 та 1) і отримують, таким чином, ключ.

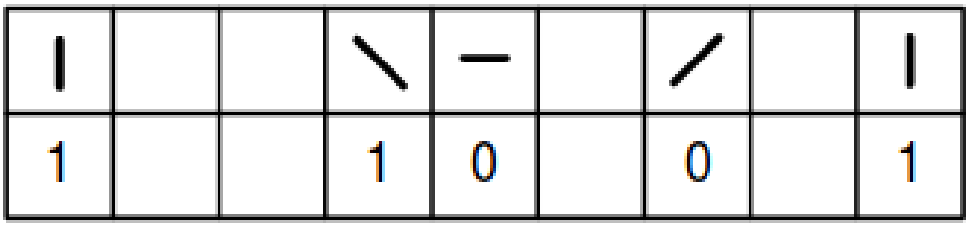


Рисунок 1.6 - Отримання ключової послідовності по результатам правильних вимірювань

Число випадків, в яких обрані базиси збіглися, буде складати в середньому половину довжини вихідної послідовності, тобто $n = {}^1/_2$. Таким чином, в результаті передачі ключа Бобом, у випадку відсутності перешкод і спотворень будуть правильно зареєстровані у середньому 50% фотонів.

Однак, ідеальних каналів зв’язку не існує і для формування секретного ключа необхідно провести додаткові процедури пошуку помилок і підсилення секретності, які називають пост-обробкою. Головна мета пост-обробки — це створення бітів з надісланих фотонів та корекція помилок. Тому важливо визначити джерело появи помилок. Ненадійний квантовий канал має таку назву через те, що при транспортуванні фотонів можуть виникати шуми, які змінюють вектор поляризації фотона. Також можуть виникати помилки при прийнятті фотонів Бобом та помилкове зчитування стану фотона. Однофотонний детектор, який є надзвичайно чутливим елементом для виявлення одиничних фотонів, неминуче сприймає деякі шуми, такі як Dark count, After Pulse, а також Cross talk з інших каналів [28]. Для протоколу BB84 його авторами встановлена допустима відносна кількість квантових помилок (quantum bit error rate, QBER), яка складає менше 11%. В такому випадку користувачі з нерозкритої послідовності після корекції помилок через відкритий канал зв’язку і посилення

конфіденційності, можуть отримати секретний ключ, який буде однаковий для них і не буде відомий третій стороні (в подальшому – Єва). Ключ, отриманий до корекції помилок і підсилення конфіденційності, називається просіяним ключем. Зазвичай, QBER від усереднених фонових шумів системи в ненадійному квантовому каналі становить близько 5%, які необхідно виправити. Саме тому використовують корекцію помилок на етапі пост-обробки. Також слід зазначити, що на цьому та на наступних етапах обмін інформацію між Алісою і Бобом відбувається по класичному каналу, який з великою долею вірогідності можна вважати надійним. Тому задача стоїть саме у виправленні помилок, які виникли на етапі обміну фотонами по ненадійному квантовому каналу, хоча в реальності неможливо визначити помилки, які виникли через перешкоди або через вплив третьої сторони.

1.5. Поля Галуа та методи корекції помилок

1.5.1. Поля Галуа

Поля Галуа, названі в честь видатного математика Евариста Галуа, який започаткував основи сучасної математики в цілому та теорію скінчених полів зокрема (які ще називають скінченими полями), знайшли своє використання в цифрових комунікаційних системах у трьох основних напрямках: кодування з виявленням помилок, кодування з виправленням помилок, а також формування псевдовипадкових послідовностей [29].

Поле, в якому скінчена кількість елементів, називають скінченим полем або полем Галуа. Позначається воно як $GF(q)$, де q число елементів поля, що називається порядком поля. В будь-якому полі існують лише операції додавання та множення. В полях Галуа існують одиничний елемент відносно операції додавання, який прийнято позначати символом “0” і називати нулем, та

одиничний елемент в полі відносно операції множення, який прийнято позначати символом “1” і називати одиницею.

Так як комп’ютери та телекомунікаційні системи оперують бітами, найбільшу увагу отримує саме $GF(2)$, в якому зворотній адитивний елемент до елемента 1 є елемент 1, тобто -1 = 1. Зворотнім мультиплікативним елементом до елемента 1 є також елемент 1, так як 1×1 = 1.

1.5.2. Векторні простори над полями Галуа

Розглянемо деяку вільну множину V, яка називається векторним простором, а його елементи — векторами.

Вектори, лінійними комбінаціями яких може бути представлений будь-який вектор векторного простору V, називаються векторами, що породжують векторний простір V. Будь-який вектор векторного простору може бути представлений у вигляді лінійної комбінації базисних векторів:

$$v = a_1e_1 + a_2e_2 + ... + a_ne_n, \qquad (1.1)$$

де v — довільний вектор векторного простору V, $e_1, e_2, ..., e_n$ — базисні вектори векторного простору V, $a_1, a_2, ..., a_n$ — скаляри деякого скалярного поля F.

Базис вигляду

1 0 0 ... 0,
0 1 0 ... 0,
0 0 1 ... 0,
- - - - - - -
0 0 0 ... 1.

називається стандартним базисом [30].

1.5.3. Методи корекції бітових помилок

Методи корекції помилок, які використовуються в сучасних цифрових система зв’язку, діляться на два великі класи: методи зворотного виправлення помилок (Backward Error Correction, BEC) і методи прямого виправлення помилок (Forward Error Correction, FEC). Як відомо, сучасні технології цифрового зв’язку в процесі передачі ділять дані на фрагменти, які називають кадрами або фреймами. В основі зворотного методу виправлення помилок завжди лежить деякий метод перевірки помилок, які виникли в процесі передачі кадра. У випадку виявлення помилок у певному кадрі він передається знову. Такий метод передбачає використання команд керування повторною передачею між відправником та отримувачем. Методи зворотного виправлення помилок достатньо прості для розуміння і апаратної реалізації. В цьому їх головна перевага. Однак, застосування цих методів приводить до зниження фактичної пропускної здатності каналів зв’язку за рахунок росту трафіку, зв’язаного з повторною передачею кадрів. В цьому полягає головний недолік методів BEC.

Функція прямого виправлення помилок заснована на специфічній обробці кадру, в результаті якої помилки всього кадру або окремої його частини можуть бути виправлені самим приймачем повідомлення. Пряме виправлення помилок володіє більшою універсальністю. В основі функції FEC знаходяться коди, які виправляють помилки. Такі коди використовують алгебраїчні методи теорії полів Галуа. Більшість кодів FEC є блоковими кодами. Для забезпечення можливості виправлення помилок вихідний блок даних деякої довжини k перетворюється по деякому правилу в кодове слово довжини n ($n > k$). Розширення довжини вихідної послідовності використовується для знаходження та виправлення помилок. Іншими словами, кодове слово c — це послідовність довжини n, яка утворена з вихідної послідовності довжини k по деякому правилу [29].

1.5.3.1. Лінійні коди

Нехай C — лінійний (n, k)-код над $GF(q)$, тобто q-ковий лінійний (n, k)-код. Нехай $g_1, g_2, ..., g_k$ — базисні вектори простору $GF^k(q)$ коду C. Будь-який кодовий вектор може бути представлений у вигляді лінійної комбінації базисних векторів коду C з коефіцієнтами з $GF(q)$. І навпаки, будь-яка лінійна комбінація базисних векторів коду C з коефіцієнтами з $GF(q)$ є кодовим вектором. Оскільки кожен коефіцієнт лінійної комбінації базисних векторів $g_1, g_2, ..., g_k$ як елемент $GF(q)$, може незалежно від інших коефіцієнтів приймати q різних значень, то всього існує q^k таких лінійних комбінацій. Нехай i — деяка q-кова інформаційна послідовність довжини не більше k символів. Така послідовність може бути представлена послідовністю $i = (i_1, i_2, ..., i_k)$ елементів $GF(q)$, тобто k-послідовністю елементів $GF(q)$. Всього можна визначити q^k таких послідовностей. Процес кодування полягає в однозначній заміні з q^k інформаційних послідовностей одним з кодових слів коду C [29].

Розглянемо лінійну комбінацію:

$$i_1 \times g_1 + i_2 \times g_2 + ... + i_k \times g_k. \qquad (1.2)$$

Оскільки $i_1, i_2, ..., i_k$ належать $GF(q)$, то така лінійна комбінація, очевидно, відповідає деякому вектору коду C. Так як вектори $g_1, g_2, ..., g_k$ утворюють базис коду C, то кожна з q^k можливих k-послідовностей інформаційного повідомлення i єдиним чином представляється у вигляді лінійної комбінації векторів $g_1, g_2, ..., g_k$, тобто у вигляді єдиного кодового слова c коду C. Таким чином, q^k усіх різноманітних q-кових інформаційних повідомлень єдиним образом відображаються в q^k -послідовностей кодових слів лінійного кода C. Таким чином, визначено кодове слово c як лінійну комбінацію скалярів поля і базисних векторів векторного простору коду C. Уявімо тепер послідовність кодового слова c і послідовність інформаційного повідомлення i у вигляді матриць-рядків розміру $1 \times n$ та $1 \times k$ відповідно. Тоді у відповідності з

правилами додавання та множення на скаляр векторів з $GF^n(q)$, лінійна комбінація вигляду (1.2) може бути записана у вигляді (рис. 1.7):

$$\left[i_1\, i_2 \ldots i_k\right] \cdot \begin{pmatrix} g_{11}\, g_{12} \ldots g_{1n} \\ g_{21}\, g_{22} \ldots g_{2n} \\ \cdot \quad \cdot \\ g_{k1}\, g_{k2} \ldots g_{kn} \end{pmatrix} = \left[c_1\, c_2 \ldots c_n\right].$$

Рисунок 1.7 - Утворення кодового слова

Де g_{ij}- j-я компонента i-го базисного вектора, i = 1, 2, ..., k, j = 1, 2, ..., n. Чи в більш компактній формі:

$$c = i \times G, \tag{1.3}$$

де G — матриця коефіцієнтів базисних векторів $g_1, g_2, \ldots, g_k$ коду C. Очевидно, що рядки матриці G можуть бути утворені будь-якими k лінійно незалежними n-послідовностями елементів $GF(q)$, які відповідають базису коду C. Матриця G розміру $k \times n$, простір рядків якої прирівнюється до коду C, називається породжувальною матрицею коду C [29].

1.5.3.2. Породжувальна матриця

Код, перші k символів кожного кодового слова якого відповідають вихідним інформаційним символам, називається систематичним кодом. Інші n — k символів називаються перевіряючими символами. Мірою збільшення довжини кодового слова у порівнянні з інформаційною послідовністю є надлишковість коду. Абсолютна надлишковість визначається різницею числа символів кодового і інформаційного слів: $r = n$ — k. Породжувальна матриця G

лінійного систематичного коду відповідає розширенню одиничної матриці розміру k (рис. 1.8.):

$$G = \left(\begin{array}{ccc|ccc} 1 & 0 & \dots & 0 & p_{11} & \dots & p_{1r} \\ 0 & 1 & \dots & 0 & p_{21} & \dots & p_{2r} \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & 1 & p_{k1} & \dots & p_{kr} \end{array}\right) = \left[E_k \mid P\right],$$

Рисунок 1.8 - Породжувальна матриця в систематичному вигляді

де E_k — одинична матриця порядку k, P — матриця перевіряючих символів розміру $k \times r$, де $r = n - k$ - число перевіряючих символів кодового слова.

Перетворення породжувальної матриці G з несистематичного виду до систематичного виду відбувається за допомогою матричних перетворень Гауса.

Відношення k/n називається швидкістю коду і позначається як R.

Нехай H — матриця, рядки якої відповідають базисним векторам ортогонального доповнення q-кового лінійного коду C довжини n. Тоді для будь-якого вектора c, який належить коду, справедливо:

$$c \times H^T = 0 \qquad (1.4)$$

Умова (1.4) дозволяє перевірити належність довільної n-послідовності елементів $GF(q)$ певному q-ковому лінійному коду. Якщо множина n-послідовностей утворює підпростір розмірності n, то ортогональне доповнення цього підпростору буде мати розмірність $n - k$. Підпростір розмірності $n - k$ утворюють будь-які $n - k$ базисних векторів. Тому матриця H повинна містити $n - k$ лінійно незалежних рядків. Так як матриці G і H належать одному простору n-послідовностей, то число стовпців матриці H дорівнює числу стовпців матриці G. Таким чином, матриця H має розмір $(n — k) \times n$. Результатом

перемноження матриць G розміру $k \times n$ і H^T розміру $n \times (n - k)$ є матриця розміру $k \times (n - k)$, яка складається з нульових елементів. Матриця H, рядками якої є базисні вектори ортогонального доповнення підпростору лінійного коду, називається матрицею перевірки лінійного коду [29].

1.5.3.3. Матриця перевірки

Матриця H^T є розширенням матриці $-P$ і окрім матриці $-P$ матриця H містить одиничну матрицю порядку $n - k$. Матриця H^T є результатом транспонування матриці H. Результатом повторного транспортування матриці є вихідна матриця. Тому матриця H може бути отримана шляхом транспортування матриці H^T. Так як для будь-якого елемента a поля $GF(2)$ справедливо $a = -a$, то для двійкового лінійного коде справедливо $P = -P$. Матриця двійкового лінійного коде буде мати наступний вигляд (рис. 1.9):

$$H = \left[P^T \,\vdots\, E_{n-k} \right] = \left(\begin{array}{cccc:c} p_{11} & p_{12} & \dots & p_{1k} & 1\,0\,\dots\,0 \\ p_{21} & p_{22} & \dots & p_{2k} & 0\,1\,\dots\,0 \\ & & \ddots & & \ddots \\ p_{n-k\,1} & p_{n-k\,2} & \dots & p_{n-k\,k} & 0\,0\,\dots\,1 \end{array} \right).$$

Рисунок 1.9 - Матриця перевірки

Матриця P, яка входить в склад матриці G і містить перевірочні символи, може бути отримана шляхом транспортування матриці P^T, яка входить в склад матриці H. Таким чином, будування лінійного коду зводиться до пошуку матриці H. По заданій матриці H легко можна знайти матрицю G.

1.5.3.4. Декодування лінійних кодів

Нехай $c = (c_1, c_2, ..., c_n)$ — вектор кодового слова, який передається, а $v = (v_1, v_2, ..., v_n)$ - відповідний прийнятий вектор. Вектори c і v можуть відрізнятись по причині виникнення помилок в процесі передачі кодового слова. Ця різниця може бути записана так:

$$e = v — c, \tag{1.5}$$

де $e = (e_1, e_2, ..., e_n)$ — вектор, символи якого приймають ненульові значення в помилкових розрядах і нульові — в розрядах, переданих без помилок. Вектор e називається вектором помилок.

Задача декодування — відшукати вектор помилок і по відомому вектору помилок відновити вихідне кодове слово.

Нехай H - матриця перевірки лінійного (n,k)-коду C. Тоді матриця-рядок $s(y) = y \times H^T$ розміру $1 \times k$ називається синдромом вектора y. Нехай $y = v = c + e$ — вектор прийнятого повідомлення. Тоді, $s(y) = s(e)$ і рівність нулю $s(v)$ означає належність вектора v коду C. Це в свою чергу означає відсутність помилок вектора v. Нехай z — також вектор векторного простору $GF^n(q)$. Очевидно, що $s(y) = s(z)$ тоді і тільки тоді, коли $y \times H^T = z \times H^T$ або $(y — z) \times H^T = 0$. Це означає, що вектор $y - z$ належить C. Тоді $y - z = c_k$ - деяке кодове слово коду C. Рівність синдромів векторів y і z означає належність обох векторів одному суміжному класу [29].

Метод декодування лінійних кодів за допомогою суміжних класів є загальних для усіх лінійних кодів.

1.5.4. Код Хемінга

Двійковий симетричний канал — найпростіша приближена математична модель каналу зв’язку з шумами, в якій біти передаються коректно з

вірогідністю p і помилково — з вірогідністю $q = 1 — p$, $p > q$. При цьому вважається, що помилки виникають незалежно (канал без пам'яті) [30].

У 1950 році американський математик Річард Хемінг розробив лінійний блоковий код, який назвали кодом Хемінга [31].

Нехай матриця H даного лінійного коду містить деяку кількість строк m. Таким чином, кожний стовпчик матриці H являє собою двійкову комбінацію довжини m. Якщо серед стовпців матриці H будуть присутні стовпці з однаковими значеннями — такі стовпці будуть лінійно залежними. При цьому умова попарної незалежності стовпців матриці H не буде виконуватись. Тому стовпці матриці H не повинні повторюватись. Число комбінацій, які не повторюються довжини m, має бути $2^m — 1$ (так як стовпець з усіма нулями є надлишковим і не міняє структуру коду).

Кількість стовпців матриці перевірки відповідає довжині n кодового слова, а число рядків — число перевіряючих розрядів r $(r = n - k)$. Число інформаційних символів k матриці H, яка містить m рядків: $k = n — m = 2^m — m — 1$. Двійковий код довжини $n = 2^m - 1$ з матрицею перевірки розміру $m \times 2^m - 1$ називається двійковим кодом Хемінга і дозволяє гарантовано виправити одну помилку кодового слова.

Визначення коду Хемінга зводиться до створення матриці H, яка містить m рядків, які відповідають усім ненульовим двійковим послідовностям довжини m. Стовпці матриці H несистематичного коду Хемінга можуть бути записані довільним чином. Пізніше систематичний вигляд можна легко отримати шляхом матричних перетворень Гауса матриці H.

Розглянемо деяку H матрицю двійкового несистематичного $(2^m — 1, 2^m — m - 1)$-коду Хемінга, стовпці якої розташовані в порядку зростання двійкових значень (такий код не є систематичним). Таким чином, кожен стовпець матриці H відповідає двійковому значенню на 1 більше попереднього стовпця. При цьому усі стовпці матриці будуть відповідати m десятковим числам від 1 до $m — 1$. Згідно визначення синдрому, вектора помилок і правилам матричного

перемноження, в даному випадку синдром буде визначатися сумою стовпців матриці *H*, номера яких відповідають помилковим розрядам прийнятого вектора (ненульовим розрядам вектора помилок). При цьому очевидно, що у випадку коду, який виправляє одну помилку кодового слова, синдром завжди дорівнює одному із стовпців матриці *H*, номер якого відповідає помилковому розряду прийнятого вектора. Таким чином, у випадку розглянутого двійкового несистематичного коду Хемінга синдром завжди відповідає двійковому запису номера помилкового розряду кодового слова у відповідності з умовним розташуванням стовпців матриці *H*.

Приведення матриці *H* до систематичного виду, очевидно, порушує порядок відповідності синдрому номеру помилкового розряду, однак також очевидно при цьому, що однозначність такої відповідності не порушується.

Таким чином, кожному вектору помилок (якщо присутня лише одна помилка) завжди відповідає однозначно визначений (єдиний) синдром. При цьому код гарантовано виправляє ту єдину помилку кодового слова.

1.6. Методи виправлення помилок в системах QKD

Вище було розглянуто основні підходи щодо корекції бітових помилок. Однак ці методи застосовуються для класичних протоколів передачі інформації. Тому необхідно визначитись, чи можна застосовувати коди корекції помилок в системах QKD без розповсюдження коригувальних бітів третій стороні (Єві). Існує багато методів корекції помилок, які застосовуються саме в системах QKD. Однак, серед них найбільш широкого застосування знайшли методи Cascade [32], Winnow [33] та LDPC (Low Density Parity Check) [34]. При цьому, деякі методи створені спеціально для систем QKD, інші - адаптовані з класичних методів корекції помилок.

Метод Cascade було запропоновано у 1994 році одним із авторів протоколу BB84 Жилєм Брассаром як вдосконалений метод корекції помилок, застосований у протоколі BB84.

Розглянемо детальніше алгоритм реалізації цього методу.

Перший крок - Аліса та Боб домовляються про випадкову перестановку по відкритому класичному каналу. Вони виконують цю перестановку на відповідних просіяних бітах, щоб спробувати рівномірно розподілити будь-які помилки. Потім Аліса і Боб ділять свій просіяний ключ на блоки розміром k, де k визначено таким чином, що кожен блок, ймовірно, має не більше однієї помилки, виходячи з рівня помилок, отриманих під час їх оцінки. Згідно алгоритму реалізації методу Cascade емпірично визначається оптимальний розмір блоку, який повинен бути приблизно $^{73}/_{P}$, де P являє собою розрахунковий коефіцієнт помилок. Після цього кроку обчислюється парний одиничний біт кожного блоку і відкрито публікується. Якщо біти парності Аліси та Боба співпали, то вони припускають, що в цьому блоці немає помилок і вони рухаються далі. З іншого боку, якщо парність блоку не погоджується між Алісою та Бобом, то вони виконують двійковий пошук у цьому блоці з метою виявлення єдиної бітової помилки, яку вони потім виправляють. Таким чином, максимум $(1 + \log_2 k)$ бітів парності обмінюються для кожного блоку, де є помилка, а 1 біт парності обмінюється в тих блоках, які не мають помилки. Щоб врахувати ці біти через загальнодоступний канал та мінімізувати інформацію, отриману будь-якою присутньою третьою стороною (Євою), згідно алгоритму пропонується відкинути останній біт кожного блоку та підблоку, на який був обміняний біт парності. Це називається підтримкою конфіденційності. Після закінчення цього процесу і Аліса, і Боб домовляються про всі їхні парні блоки, вони можуть бути впевнені, що всі блоки містять нуль або однакову кількість помилок. Це пов’язано з тим, що перевірка парності сама по собі не може виявити однакову кількість помилок у блоці. Тому Аліса і Боб повинні знову переставити свій новий ключ до наступного проходу. Крім того, після кожного

проходження розмір блоку збільшується, щоб врахувати той факт, що залишається менше помилок.

Таким чином, Алісі та Бобу не потрібно скидати стільки бітів парності, як коли б вони виконували повний пропуск протоколу, хоча вони все ще відкидають останній біт від кожного блоку та підблоку, в якому було обміняно біт парності. В якийсь момент Аліса і Боб виявлять, що всі їх порівняння на парності узгоджуються. Коли це відбувається протягом декількох пропусків (автори пропонують 20), Аліса та Боб роблять висновок, що їх узгоджені ключі однакові і вони переходять до посилення конфіденційності.

Згідно алгоритму реалізації методу Cascade перший пропуск здійснюється діленням просіяного ключа на розміри блоків довжини k на основі оціненого коефіцієнта помилок і біти парності для кожного блоку обмінюються. Двійковий пошук виконується для того, щоб виявити поодинокі бітові помилки на блоках, що мають невідповідні паритети. Однак, жоден біт не відкидається під час цього першого проходу. Натомість помилки блоку виправляються, застосовується перестановка, розмір блоку збільшується до 2k, а інший прохід виконується ідентичним першому. Для будь-яких помилок, виправлених під час другого проходу, повинна бути принаймні одна помилка узгодження, яка знаходилась у тому самому блоці в попередньому проході, оскільки жодна помилка не була знайдена і не виправлена в цьому проході. З цієї причини для кожного виправлення, здійсненого в будь-якому проході після попереднього проходу, на блоці, що містить цей виправлений біт, у всіх попередніх проходах повторюється двійковий пошук з метою виявлення будь-яких потенційних помилок відповідності. Щоразу, коли виявляється нова помилка, вона виявляє потенціал маскування чергової помилки в попередньому проході, тому процес повторюється, а виявлення та виправлення помилок каскадується через усі попередні проходи (рис. 1.10).

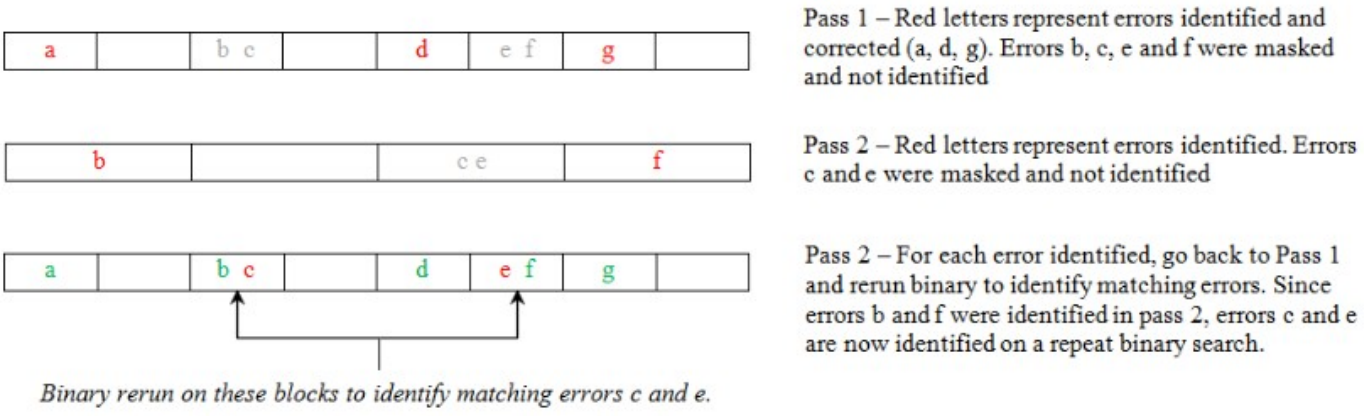


Рисунок 1.10 - Сутність алгоритму реалізації методу Cascade

У кожному проході після першого підпроходу в середньому будуть виправлені дві помилки на кожен виявлений біт, тому кількість помилок, присутніх в узгодженому ключі, зменшується експоненціально для кожного проходу. Емпірично для реалістичних показників помилок чотири проходи, як правило, вважаються достатніми для виправлення всіх помилок. Найбільшою проблемою для методу Cascade є, як правило, необхідна кількість взаємодії. Хоча сам метод Cascade виявився дуже ефективним, здатним виправляти всі помилки і не генерувати нові, якщо їх правильно реалізовувати, проте він страждає від високої швидкості інтерактивності через необхідні обміни паритету. Крім того, кількість інформації, яка обмінюється при реалізації методу Cascade, залежить від розподілу помилок, тому його пропускна здатність може бути непередбачуваною на практиці.

Метод Winnow схожий з методом Cascade. Він був розроблений у 2003 році та на відміну від Cascade він розбиває двійкові рядки на блоки, але замість виправлення помилок за допомогою ітеративної бінарної фіксації помилок метод базується на кодах Хемінга. Розглянемо принцип його роботи.

Спочатку Аліса та Боб ділять свої повідомлення на блоки довжиною k (пропонується починати з $k = 8$). Потім Аліса і Боб порівнюють паритети і відзначають будь-які розбіжності, як і в алгоритмі реалізації методу Cascade.

Різниця полягає в подальших діях. Замість того, щоб здійснювати двійковий пошук, Аліса і Боб будують матрицю перевірки парності для коду Хеммінга: H = (j / 2^{i-1}) mod 2 і обчислюють синдроми s для кожного блоку B, який не є погодженим, де $s = H \times B$. Потім Аліса надсилає свої синдроми Бобу і він, порівнюючи ці синдроми з його власними, може виправити будь-які єдині бітові помилки. Для того, щоб врахувати інформацію, яка піддається обміну паритетом та синдромом, пропонується здійснювати підтримку конфіденційності протягом усієї фази виправлення помилок. Розробники методу Winnow стверджують, що він перевершує метод Cascade, оскільки деякі біти, відкинуті на цьому шляху, насправді можуть бути помилковими бітами помилково. Отже, один біт видаляється для кожного біта паритету / синдрому, що обмінюється, хоча вибір бітів синдрому не є випадковим. Запропоновано видалення бітів у кожному блоці в положенні 2^j для $j \in \{0, ..., m-1\}$, оскільки ці біти не залежать в обчисленнях синдрому і тому найбільш піддаються впливу. Оскільки перевірки парності, що використовуються таким чином, можуть виявити лише непарну кількість помилок, після одного проходу, ймовірно, деякі блоки все ще містять парну кількість помилок. З цієї причини Аліса і Боб переставляють узгоджені довжини блоків і повторюють процес, хоча розмір блоку збільшується і матриця перевірки регенерується для врахування нового розміру блоку. Оптимальна кількість проходів та розміри блоків для кожного проходу є предметом дискусії. Якщо згідно методу Cascade алгоритм не виявить рівну кількість помилок і виправить лише одну помилку в блоках, що містять більше двох помилок (з непарним паритетом), алгоритм реалізації методу Winnow може насправді ввести помилки, якщо їх кількість на один блок є занадто великою. Причиною цього є те, що коли кількість помилок зростає, також збільшується ймовірність декодування кодового слова в неправильне кодове слово, як згадувалося раніше. З цієї причини точне оцінювання помилок та розподіл помилок є важливою складовою методу Winnow.

Найбільше обмеження методів Cascade та Winnow полягає в тому, що в алгоритмах реалізації їх обох виправляється лише одна помилка на блок. Цей факт зумовлює необхідність складної процедури перетасування, яка повинна бути виконана однаково з обох сторін між проходами, що лише додає комунікаційних витрат. Окрім цього, реалізація обох методів потребує занадто багато часу для виконання повторних ітерацій. Так як обмін бітами відбувається постійно, необхідно враховувати відстань між сторонами та довжину кодових слів. Для великих відстаней та довгих кодових слів обидва розглянуті методи є неефективними.

1.7. Метод корекції помилок LDPC та постановка задачі дослідження

Усунення зазначених вище недоліків, а саме підвищення швидкодії при можливості оперування з кодовими словами більшої довжини і на великих відстанях досягається за допомогою методу корекції помилок LDPC.

Метод було запропоновано Р. Галагером у 1960 році. Як і код Хемінга, код LDPC - це код FEC, визначений матрицею перевірки H та породжувальною матрицею G. У кодах LDPC, як і в кодах Хемінга, мінімальна відстань коду є важливим параметром, оскільки вона визначає межу його декодування. На жаль, для кодів великої довжини пошук цієї межі не є простим. Тому більш простим підходом до окремих кодів є емпіричне визначення діапазону помилок, що декодуються. У їх первісному вигляді Р. Галагер описує коди LDPC як фіксовану кількість j для “1” у кожному рядку та фіксовану кількість k для “1” у кожному стовпчику. Поряд із довжиною блоку n такий код відомий як (n, j, k) код низької щільності. Кількість “1” може бути розповсюджена випадковим чином з урахуванням обмежень: $n \times k = m \times j$. Швидкість коду r ($0 \leq r \leq 1$), як правило, заздалегідь визначається і матиме значний вплив на коригувальну

потужність та ефективність коду. Розміри породжувальної матриці та матриці корекції задаються розмірами $m \times n$, де m визначається як:

$$m = n\,(1 — r). \tag{1.6}$$

Зручним способом візуалізації коду LDPC є представлення матриці перевірки у вигляді графа Танера, що є двостороннім графом, складеним з вузлів та ребер. Кожен рядок матриці перевірки являє собою вузол перевірки, а кожен стовпець — вузол вихідного повідомлення. Кожен вузол перевірки являє собою перевірку парності, виконану під час обчислення синдрому, а кожен вузол вихідного повідомлення являє собою один біт вхідного повідомлення. Тому матриця перевірки парності $m \times n$ містить m контрольних вузлів і n змінних вузлів. Для кожного ненульового запису в положенні $[i, j]$ в матриці перевірки у графі з’являється зв’язок, що з'єднує контрольний вузол i та змінний вузол j. Кількість цих ребер, з'єднаних із заданим вузлом, є ступенем цього вузла.

У своїй роботі Р. Галагер представив алгоритм для генерування матриць LDPC за допомогою генератора псевдовипадкових чисел. Ця процедура призводить до кодів з досить широким діапазоном, однак, є обчислювально неефективною для великих розмірів ключів і застосовується лише для звичайних кодів. Саме тому в свій час метод корекції LDPC не отримав належної уваги через складність створення обчислювального пристрою для кодування та декодування повідомлень.

Однак, У 1995 році Девід Маккей та Редфорд Ніл у своїй роботі повторно відкрили LDPC коди [35], що дозволило їм через кілька років запропонувати алгоритм, який забезпечує створення матриць корекції помилок, формування з них породжувальних матриць, кодування повідомлень з інформаційних повідомлень, виправлення помилок прийнятих векторів та декодування виправлених векторів в інформаційні повідомлення [36].

Таким чином, із урахуванням зазначеного вище можна зробити висновок про те, що LDPC коди можуть бути застосовані в системах QKD на етапі

корекції помилок. Підтвердженням цього стала низка публікацій, в яких запропоновано впровадження методу LDPC у системи QKD [24], [37]–[39].

Однак, вищевказані роботи або є чисто теоретичними без експериментального підтвердження запропонованих авторами рішень, або результати впровадження методу виправлення помилок LDPC у системи QKD викликають низку питань щодо забезпечення конфіденційності, зокрема, захисту даних від третьої сторони (Єви).

Тому важливим і актуальним є вирішення науково-прикладного завдання підвищення надійності та захищеності систем QKD шляхом подальшого розвитку методу узгодження ключа із застосуванням корекції помилок на основі LDPC кодів та розроблення алгоритмічних і програмних рішень його реалізації. Ці питання детально висвітлюються в подальших розділах цієї роботи.

РОЗДІЛ 2. ОБҐРУНТУВАННЯ ВИБОРУ БАЗОВОГО МЕТОДУ ВАЛЕНТИ ВИПРАВЛЕННЯ ПОМИЛОК У СИСТЕМАХ QKD ТА ЙОГО МОДЕЛЮВАННЯ

Як вже зазначалося вище в підрозд. 1.7, LDPC коди знайшли своє застосування в методах корекції помилок у системах QKD як альтернатива методам Cascade та Winnow. Проте при цьому виникає питання, яким чином застосовувати LDPC коди, щоб при виправленні помилок захистити інформацію від третьої сторони (Єви), яка може прослуховувати перші дві (Алісу та Боба).

2.1. Порівняльний аналіз методів Міліцевича та Валенти корекції помилок в системах QKD

Спочатку розглянемо функціонування типової системи QKD, узагальнена структура якої наведена на рис. 2.1. На першому етапі відбувається обмін фотонами між Алісою та Бобом по ненадійному квантовому каналу, який має таку назву через те, що при транспортуванні фотонів через канал можуть виникати шуми, які змінюють вектор поляризації фотона. Також можуть виникати помилки при прийнятті фотонів Бобом та помилкове зчитування ним стану фотона. Однофотонний детектор, який є надзвичайно чутливим елементом для виявлення одиночних фотонів, неминуче сприймає шуми, такі як Dark count, After Pulse, а також Cross talk з інших каналів. Беннет і Брассар визначили, якщо квантовий коефіцієнт бітових помилок (quantum bit error rate, QBER) складає менше 11%, рейтинг секретного ключа (secret key rate, SKR) становить майже нуль, тобто можна вважати Єву відсутньою і секретний ключ вважається безпечним. Для сучасних систем QKD QBER не перевищує 5%. Тобто, за умови, якщо новоутворені ключі в результаті відрізняються більше, ніж на 11% (при визначених попередньо 5%), Аліса та Боб припускають, що присутня Єва та розривають зв’язок. Якщо QBER знаходиться в межах 5%, Аліса та Єва вважають, що Єва відсутня, а отримані помилки виникли через вказані вище перешкоди в квантовому каналі, які необхідно виправити.

Визначення присутності Єви не входить в завдання даної дисертаційної роботи, отже, від зараз і надалі вважатимемо, що помилки, які виникли в просіяному ключі, є результатом лише виникнення перешкод у квантовому каналі.

Перший протокол BB84, розглянутий в розд. 1, є системою QKD з дискретними змінними (discrete variable QKD, DV-QKD) [40].

Рисунок 2.1 - Узагальнена структура системи QKD

Історично DV-QKD системи розповсюджувались швидше і наразі є більш поширеними та дослідженими. В системах DV-QKD носіями інформації на першому етапі є фотони. Аліса кодує свою інформацію в поляризації однофотонних станів. Секретний ключ встановлюється при виявленні окремих фотонів. Боб в свою чергу використовує однофотонний детектор, який вимірює кожний отриманий квантовий стан фотону. На сьогодні системи DV-QKD досліджуються протягом 30 років та здатні працювати на сотні кілометрів. Однак, для визначення квантових станів фотонів детектори мають працювати на кріогенних температурах, що є недоліком даної системи. В цих дослідженнях акцент ставиться загалом на перший етап функціонування структури, але етапу корекції помилок, на жаль, не приділено достатньої уваги.

Окрім систем DV-QKD на даний момент поширені також системи QKD з неперервними змінними (countinuous varuable QKD, CV-QKD). В цих рішеннях Аліса кодує свою інформацію в амплітудній та фазовій квадратурах когерентних станів. Системи CV-QKD можуть бути реалізовані за допомогою стандартних економічно ефективних детекторів, які, зазвичай, використовуються в класичному телекомунікаційному виробництві та працюють при кімнатній температурі. Більшість досліджень систем QKD зосереджені на застосуванні оптичного волокна, оскільки квантові сигнали як для CV-, так і для DV-QKD систем можуть використовуватись для класичного телекомунікаційного трафіку в існуючих волоконно-оптичних мережах.

Розглядаючи ті чи інші переваги та недоліки тієї чи іншої системи на різних етапах, через більш детальне дослідження DV-QKD систем на першому етапі, вибір базового методу корекції помилок в подальшому здійснено в роботі саме з орієнтацією на використання в системах DV-QKD з подальшим вдосконаленням етапу підсилення конфіденційності, в результаті чого буде запропоновано завершений, повноцінно реалізований і працездатний QKD протокол.

В 2017 році Міліцевич та інші дослідники запропонували використання LDPC матриць для CV-QKD систем [39]. Це дало змогу розширити максимальну відстань узгодження ключів між двома віддаленими сторонами. В їх дослідженнях показано, що узгодження ключів у кількох вимірах сприяє покращенню виправлень помилок багатовимірних LDPC-кодів у системах CV-QKD. Тим самим збільшено як таємну швидкість ключа, так і відстань. Однак, обчислювальна складність LDPC-декодування для великої довжини блоку порядку 10^6 біт залишається складним завданням. Тому було застосовано квазіциклічну (quasi-cyclic, QC) [41] кодову конструкцію для багатовимірних кодів LDPC, що дуже підходить для програмного забезпечення прискореного декодування на сучасних графічних процесорах (GPU). При цьому процес захисту інформації від Єви на етапі корекції помилок полягає в тому, що одна із сторін генерує інформаційний вектор абсолютно випадковим чином, який надалі буде йти на етап підсилення конфіденційності та кодується завдяки QC-LDPC в кодове слово. Після цього створюється публічне повідомлення для другої сторони шляхом математичного обчислення просіяного ключа та кодового слова. В даному випадку виходить вільне перевтілення шифру Вернама, а просіяний ключ є ключем даного шифру. Після пересилання публічного повідомлення до іншої сторони тим же шляхом математичного обчислення з’являється вектор, який треба декодувати LDPC кодом, але вже який містить помилки між просіяними ключами Аліси та Боба. Після декодування обидві сторони матимуть однакові інформаційні вектори, які будуть надіслані на наступний етап підсилення конфіденційності. Даний метод корекції помилок, запропонований Міліцевичем, є одним з прикладів можливого виправлення помилок та забезпечення захищеності просіяних ключів між Алісою і Бобом, які може перехопити Єва.

В 2013 році дослідник Мартінес-Матео [37] та у 2014 дослідник Валента [38] незалежно один від одного провели дослідження по впровадженню QC-LDPC кодів в системи DV-QKD.

Розглянемо детальніше роботу методу Валенти, автор якого в своєму дослідженні зробив акцент на підвищення швидкості та збільшення дальності роботи системи, а також можливості застосування у оптичних волокнах. В результаті було представлено повноцінну і завершену на усіх етапах QKD систему.

Виникає питання, яким чином відбувається захист даних під час обміну інформацією про помилкові біти у запропонованій Валентою QKD системі?

На етапі корекції помилок Валента пропонує розділити просіяний ключ на декілька частин та створювати синдроми з кожної частини, які передаються від Аліси до Боба. Перехопивши синдром, Єва не знає, що далі робити з ним, бо не може його декодувати (можна провести аналогію, що синдром є хеш-функцією). Боб створює з його частини просіяного ключа синдром та порівнює з отриманим синдромом. Якщо синдроми однакові, Боб зберігає поточну частину просіяного ключа, якщо синдроми різні — відкидає. Як бачимо, Боб лише зберігає чи відкидає ту чи іншу частину просіяного ключа, тобто він не декодує синдром. Тож можна зробити висновок, що в цьому випадку довжина просіяного ключа може бути достатньо довгою, але довжина окремої частини має бути достатньо короткою, щоб містити меншу кількість помилкових біт в кожній частині. Слід зазначити, що розглянутий метод Валенти відрізняється від запропонованого Міліцевічем методу корекції та дещо схожий з методом корекції помилок Winnow, який теж використовує синдром.

Алгоритм реалізації запропонованого Валентою методу корекції помилок наведено на Рис. 2.2.

Розглянуті вище методи Валенти та Міліцевича мають свої переваги та недоліки, але в результаті проведеного порівняльного аналізу в якості базового для подальших досліджень і удосконалень в роботі обґрунтовано вибір саме методу Валенти по наступним причинам:

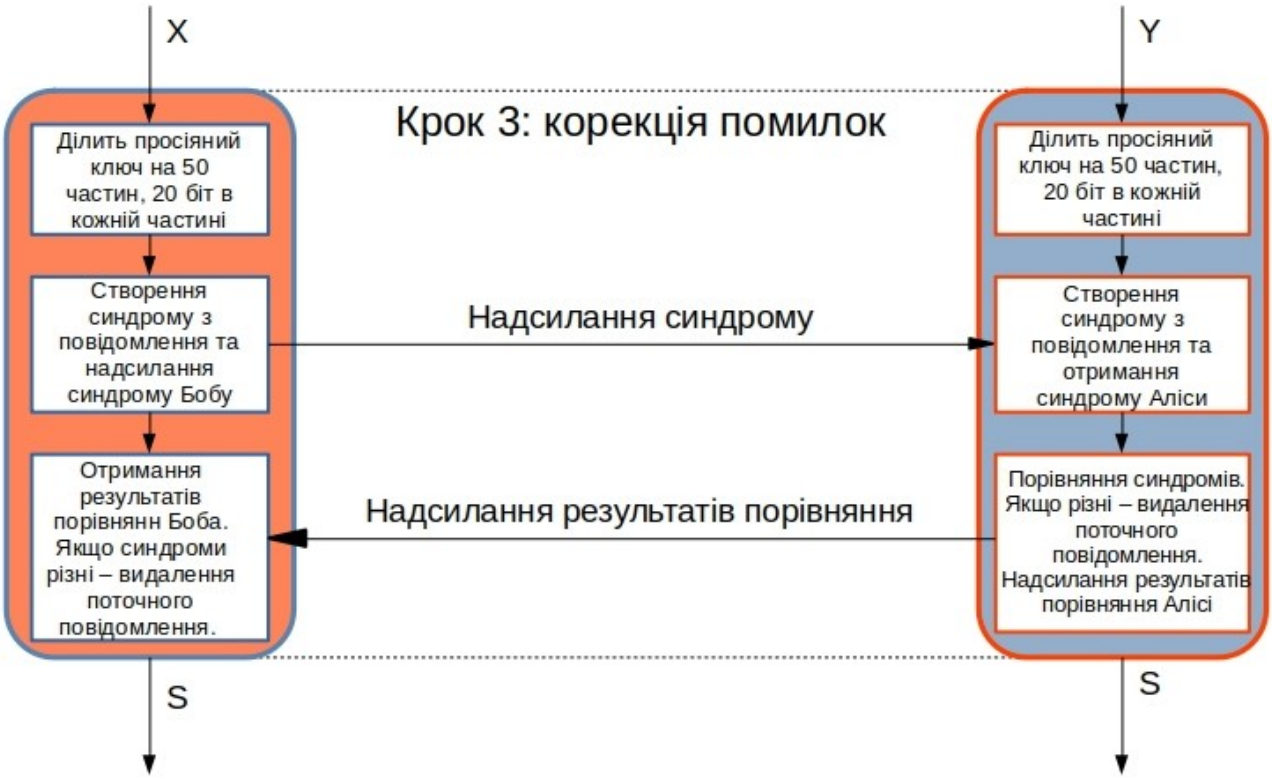


Рисунок 2.2 - Алгоритм корекції помилок на основі методу Валенти

- Метод Валенти вже застосовується в DV-QKD системах, в той час як метод Міліцевича застосовується в CV-QKD системах та має бути адаптований до DV-QKD систем.
- Метод Валенти використовує лише матрицю перевірки, в той час як метод Міліцевича потребує використання як матриці перевірки, так і породжувальної матриці.
- Метод Валенти використовує звичайну операцію порівняння, в той час як метод Міліцевича використовує складні математичні обчислення. В цьому випадку, навіть якщо довжина ключа Валенти буде набагато довша за рахунок того, що буде відкинута велика кількість частин ключа в майбутньому, час на здійснення цієї операції в кінцевому результаті буде менше, ніж для обчислення формул при використанні метода Міліцевича.

Розглянемо детальніше особливості реалізації обраного методу Валенти.

2.2. Дослідження процедури корекції помилок методом Валенти

Для проведення дослідження запропоновано генерувати випадковим чином повідомлення довжиною 1000 біт та змінювати випадковим чином окремі біти до обраного QBER. В результаті матимемо два повідомлення, які відрізняються деякою кількістю бітів, розташованих випадковим чином. Ці повідомлення вважаються просіяними ключами Аліси та Боба. Далі здійснюємо поділ просіяних ключів на декілька повідомлень фіксованої довжини. Пропонується ділити просіяні ключі на 50 частин по 20 біт кожна. Інші варіації та причини вибору саме 50 повідомлень будуть розкриті нижче.

Для обміну інформацією пропонується використовувати синдроми фіксованої довжини. Якщо довжина синдрому буде рівною довжині повідомлення, то Єва може легко, маючи матрицю H шляхом зворотного матричного перемноження відтворити повідомлення від Аліси. Однак, якщо використовувати довжину синдрому меншу, ніж довжину повідомлення, виникають ситуації, коли одному синдрому відповідають декілька повідомлень (через властивості полів Галуа та векторних просторів над ними). В такому випадку необхідно знайти таке співвідношення довжини повідомлення і синдрому, щоб кількість різних повідомлень на однаковий синдром не була занадто великою, при цьому Єва не змогла з отриманого синдрому відтворити оригінальне повідомлення. В наших дослідженнях пропонується довжина синдрому в 2 рази менше за довжину повідомлення, тобто 10 біт. Такий вибір пояснюється тим, що швидкість кодування $^1/_2$ є стандартною та широко застосовується. Отже, в результаті генеруємо матрицю перевірки H розміром 10×20. Як було вказано вище, знаходити породжувальну матрицю не потрібно. В результаті, можемо мати $2^{20} = 1048576$ різних комбінацій повідомлень та $2^{10} = 1024$ різних варіацій синдрому. Кожен синдром матиме $^{1048576}/_{1024} = 1024$ однакових повідомлення. Тобто, можлива ситуація, коли Аліса та Боб матимуть однакові синдроми при різних повідомленнях. Вірогідність такого випадку

складає $^{1}/_{1024}$ = 0,1%, що є достатньо високою. В результаті на наступний етап підсилення конфіденційності підуть різні секретні ключі, що не є допустимим. Про це йтиметься далі.

2.2.1. Генерування повідомлення із заданим коефіцієнтом квантових бітових помилок QBER

Для реальної QKD системи QBER складає 5%, але для більш широкого розуміння будемо досліджувати діапазон QBER від 0 до 25%. Так як довжина повідомлення — 1000 біт, то у випадку, коли QBER складає 5% - мають бути змінені 50 біт повідомлення, тобто 1 біт відповідає 0,1% QBER.

Генерування QBER відбувається в заздалегідь створеній програмі на мові C та відбувається за наступним алгоритмом:

1. Програма зупиняється біля кожного біта повідомлення.
2. Змінній *t* присвоюється випадкове число в діапазоні від 0 до 999.
3. Якщо *t* менше QBER — поточний біт інвертується (виникає помилка).
4. Якщо *t* більше QBER — поточний біт зберігається
5. Програма переходить до наступного біта.

Як витікає із алгоритму, кількість змінених біт, наприклад, для QBER = 5% може не бути рівною чітко 50, а може бути менше чи більше. Тому виконано повторення для кожного QBER процес генерування помилкових біт 100 раз. Наразі на надалі моделювання відбуватиметься в програмному середовищі GNU Octave. Результат наведено на рис. 2.3. Як видно, кількість змінених біт може бути більше чи менше 50, але прослідковується розподілення Пуассона, тобто можна вважати, що для майбутніх досліджень наведений вище алгоритм генерування помилок в просіяних ключах є задовільним.

На наступному етапі після генерування помилок вважаємо, що оригінальне повідомлення – просіяний ключ Аліси, повідомлення з помилками – просіяний ключ Боба.

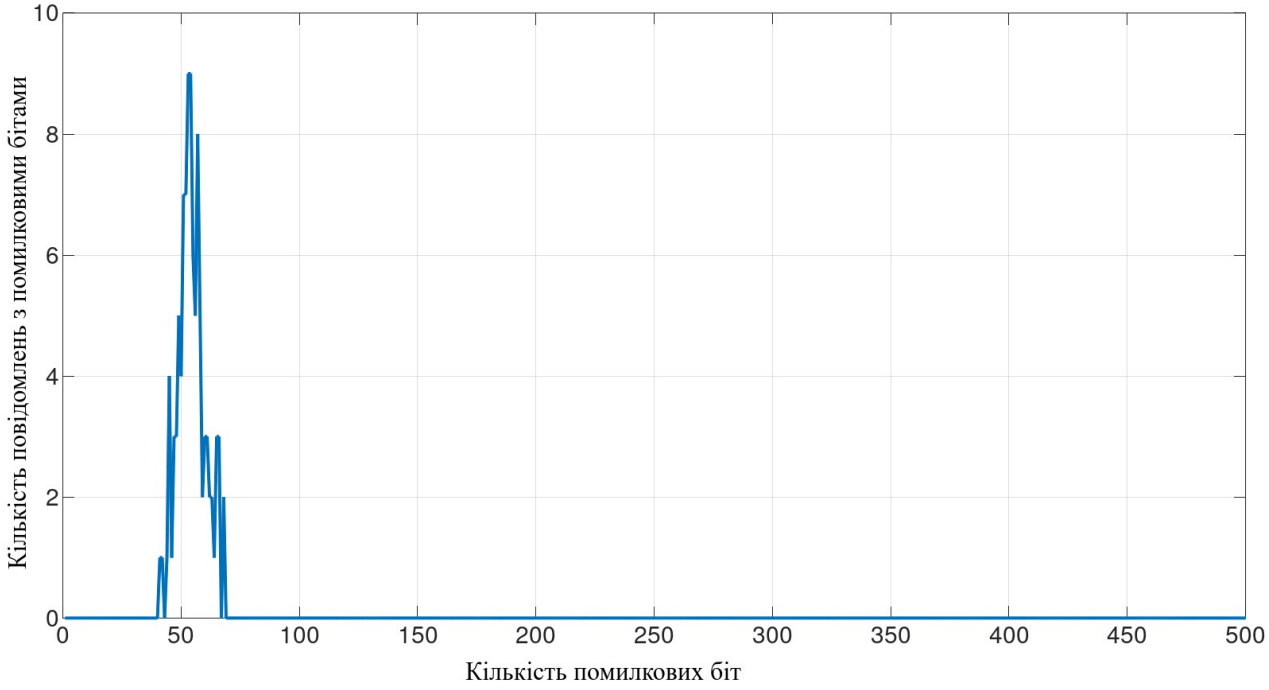


Рисунок 2.3 - Розподілення змінених біт для QBER = 5%

2.2.2. Моделювання процедури корекції помилок методом Валенти

Аліса і Боб ділять їх ключі на 50 частин (повідомлень) по 20 біт в кожній частині (повідомленні). За допомогою матричного перемноження поточного повідомлення на матрицю перевірки створюємо синдром. Необхідно нагадати, що в оригінальному застосуванні LDPC кодів синдром має показувати помилку в кодовому слові. Якщо маємо лише одну помилку, то вектор синдрому буде дорівнювати певному стовпцю матриці. Номер цього стовпця буде номером помилкового біта в кодовому слові. В методі Валенти оригінальне кодове слово є невідомим, тобто по отриманому синдрому не можемо декодувати його та віднайти кодове слово. Синдром виступає лише мірою певної інформації. Цей синдром надсилається до Боба, який створює свій синдром та порівнює його з

отриманим синдромом Аліси. Якщо синдроми однакові — зберігаємо дане повідомлення (з 50 існуючих), якщо синдроми різні — відкидаємо.

Робимо це за наступним алгоритмом:

1. Порівнюємо 50 синдромів між собою. Лічильник спільних синдромів дорівнює нулю.
2. Якщо на поточному кроці порівняння синдроми однакові — додаємо +1 до лічильника.
3. Якщо на поточному кроці порівняння синдроми різні — додаємо +0 до лічильника.

Очевидно, що максимальним числом, яке може містити лічильник, є число 50 — усі 50 повідомлень просіяних ключів є однаковими, тобто немає жодної помилки. Проведено дослідження кількості спільних синдромів для QBER від 0,1% до 25% з кроком 0,1%. Для кожного значення QBER створювалось 100 повідомлень, які ділили на 50 частин та порівнювали їх синдроми. Після цього визначалось середнє арифметичне з цих 100 повторювань для кожного QBER. Рис. 2.4. містить результат проведеного моделювання.

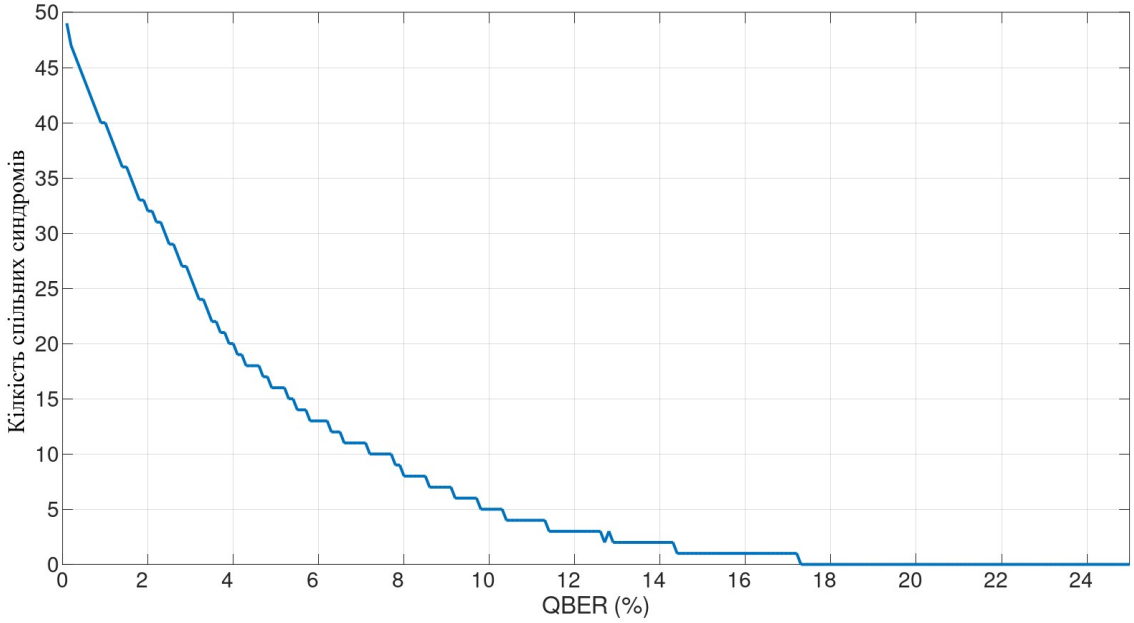


Рисунок 2.4 - Середня кількість спільних синдромів для різних QBER згідно методу Валенти

Як витікає з рис. 2.4, для QBER = 5% усереднена кількість спільних синдромів і повідомлень дорівнює 16. Тобто, з 50 повідомлень лише 16 утворять захищений ключ, інші повідомлення відкинуться і в результаті просіяний ключ з 1000 біт скоротиться до 80, що є надзвичайно малим. Звичайно, якщо адаптувати такий підхід до умов, коли можна генерувати та відкидати велику кількість біт швидше, ніж виправляти їх, але в поточних умовах необхідно саме виправляти помилкові біти, про що йтиметься в наступному розділі.

Тепер розглянемо два інші випадки: коли ділимо просіяний ключ на 100 частин, тоді в кожному повідомленні буде 10 біт, довжина синдрому складатиме 5 біт; коли ми ділимо просіяний ключ на 25 частин, тоді в кожному повідомленні буде 40 біт, довжина синдрому складатиме 20 біт. Повторимо дані дослідження для цих випадків.

На рис. 2.5 наведено результати порівняння по трьом зазначеним вище випадкам без усереднення результатів. На першому зверху графіку просіяний ключ поділений на 50 частин (усереднений графік на рис. 2.4), на другому графіку просіяний ключа поділений на 25 частин, на третьому графіку просіяний ключ поділений на 100 частин.

Очевидно, що найкращим рішенням є поділ повідомлення на 100 частин. Тоді присутня одна помилка в повідомленні буде розділена між двома повідомленнями, одне з яких відкинеться, а інше збережеться. Кількість можливих комбінацій повідомлення складатиме $2^{10} = 1024$, а кількість можливих синдромів $2^5 = 32$. І кожному синдрому будуть відповідати 32 можливі повідомлення. Тоді Єва зможе набагато легше методом підбору визначити можливі комбінації кодового слова, що є небезпечним. З іншого боку, якщо довжина повідомлення складає 40 біт, можливих комбінацій буде 2^{40}, а можливих синдромів 2^{20}. На один синдром припадатиме більше одного мільйона можливих повідомлень, що для Єви складатиме проблему для знаходження можливого кодового слова. Однак, при малому QBER, якщо в повідомленні є лише одна помилка — відкидається все повідомлення, це не є раціональним.

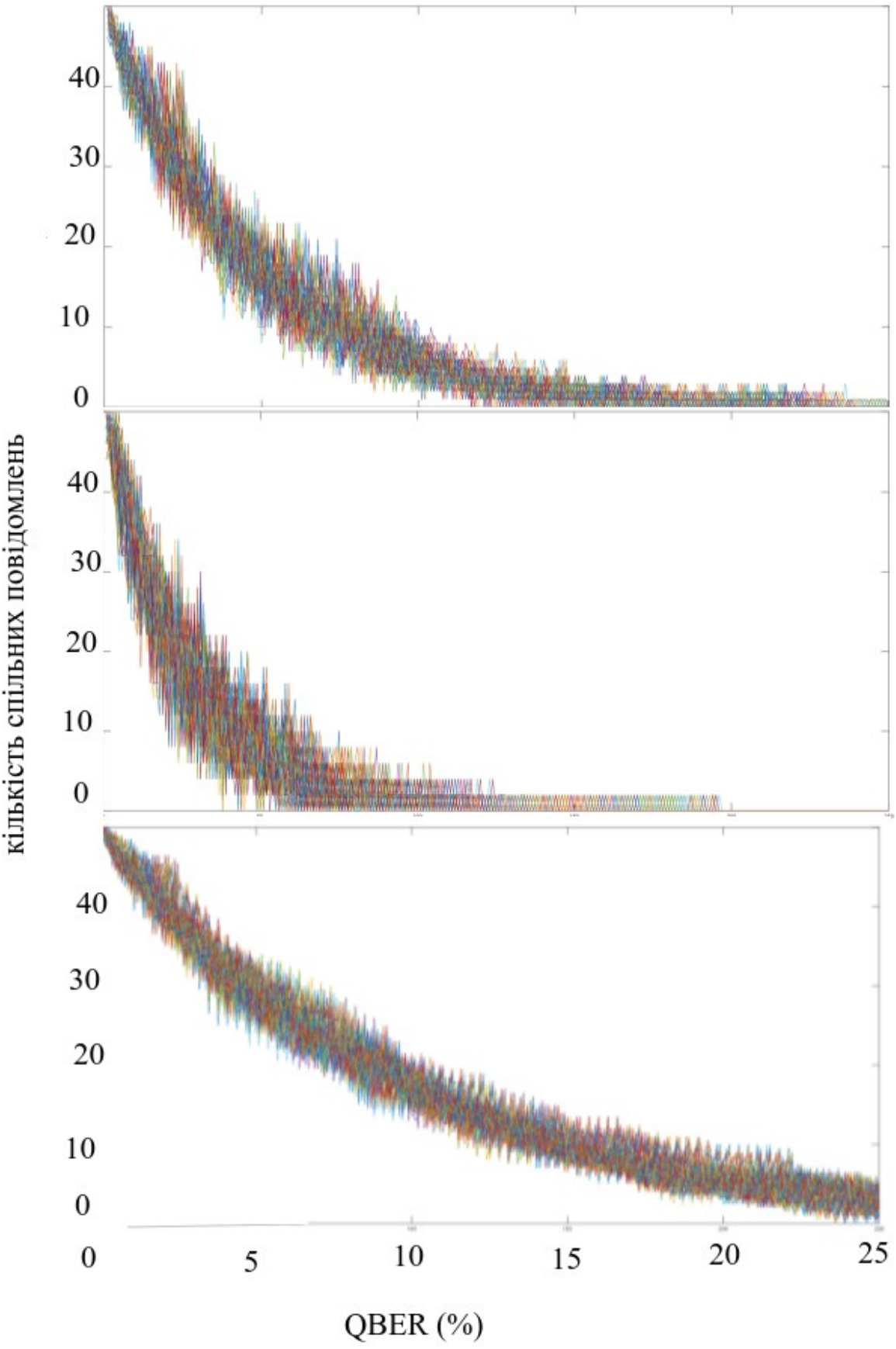


Рисунок 2.5 — Кількість спільних повідомлень для різних QBER згідно методу Валенти для трьох різних довжин повідомлень

З даної точки зору обраний від самого початку розмір повідомлень є найбільш вдалим компромісом між складністю знаходження Євою вихідного повідомлення та кількістю відкинутих повідомлень при наявності помилки.

2.3. Визначення оптимальної матриці перевірки

В отриманих 16 випадках маємо однакові синдроми. Але як було сказано вище, можливі ситуації, коли синдроми є спільними, а повідомлення різними. Для дослідження цієї колізії пропонується наступне:

1. Порівнюємо 50 синдромів. Лічильник дорівнює нулю.
2. Якщо на поточному кроці порівняння повідомлення однакові — додаємо +1 до лічильника.
3. Якщо на поточному кроці порівняння повідомлення різні — порівнюємо синдроми Аліси та Боба.
4. Якщо синдроми теж різні — додаємо +1 до лічильника.
5. Якщо синдромі однакові (повідомлення при цьому різні) — додаємо +0.

Очевидно, що знову максимальним значенням лічильника може бути 50. Для проведення дослідження згенеровано матрицю перевірки методом, запропонованим Маккеєм та Нілом. Для генерації матриці використовувалось програмне забезпечення, створене Нілом для роботи з LDPC кодами, яке знаходиться у відкритому доступі [42]. Обрано найбільш можливу розріджену матрицю, в якій у кожному стовпці наявна лише одна “1”, усі інші “0” (при цьому очевидно, що в матриці будуть 10 пар однакових стовпців).

В дослідженні знову розглядаємо випадки для QBER від 0,1% до 25% з кроком 0,1%. Для кожного значення QBER створюємо 100 повідомлень та порівнюємо їх. Якщо результат менше 50 — отже, в даному просіяному ключі виявились спільні синдроми при різних повідомленнях. Рис. 2.6. демонструє графік кількості спільних синдромів.

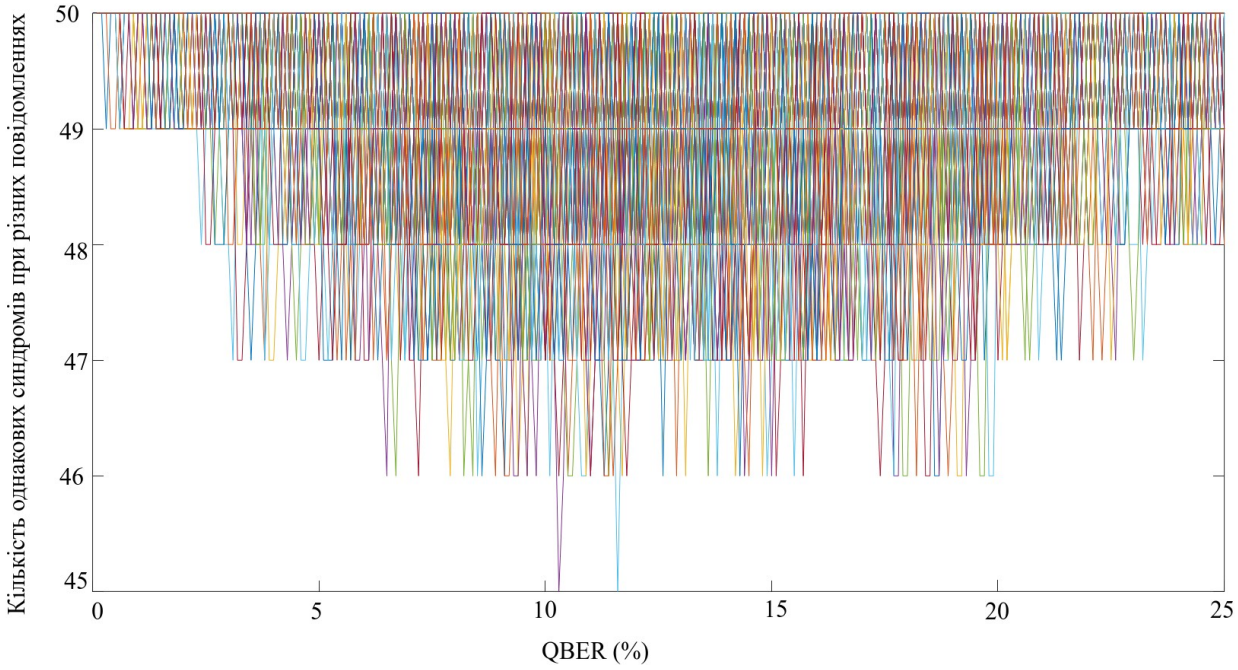


Рисунок 2.6 - Кількість однакових синдромів при різних повідомленнях для різних QBER для матриці з однією “1” в кожному стовпці

З графіку видно, що однакові синдроми при різних повідомленнях трапляються досить часто, що не є припустимим.

Після отримання результатів було висунуто припущення, що при правильному підборі параметрів можна знайти таку матрицю, при якій кількість спільних синдромів при різних повідомленнях буде відсутня.

За визначенням Галагера матриця перевірки має бути розрідженою, тобто мати дуже малу кількість “1”. Слідуючи цій логіці зроблено припущення: якщо кожен стовпець містить 10 біт, то кількість “1” в кожному стовпці має бути від одного (найменша кількість) до п’яти (найбільша кількість, половина усіх бітів). При цьому матриця може бути не систематичною. Відповідно в кожному рядку кількість одиниць буде в два рази більше.

Програмне забезпечення Ніла дозволяє генерувати матриці із заданою кількістю одиниць в стовпці та рядку процедурами “evencol” та “evenboth” з виправленням циклів розміру 4 чи ні.

В результаті розглядання усіх можливих варіантів визначено, що матриця з чотирма “1” в кожному стовпці побудована методом “evenboth” без виправлення циклів розміру 4 дає найкращі результати (рис. 2.7).

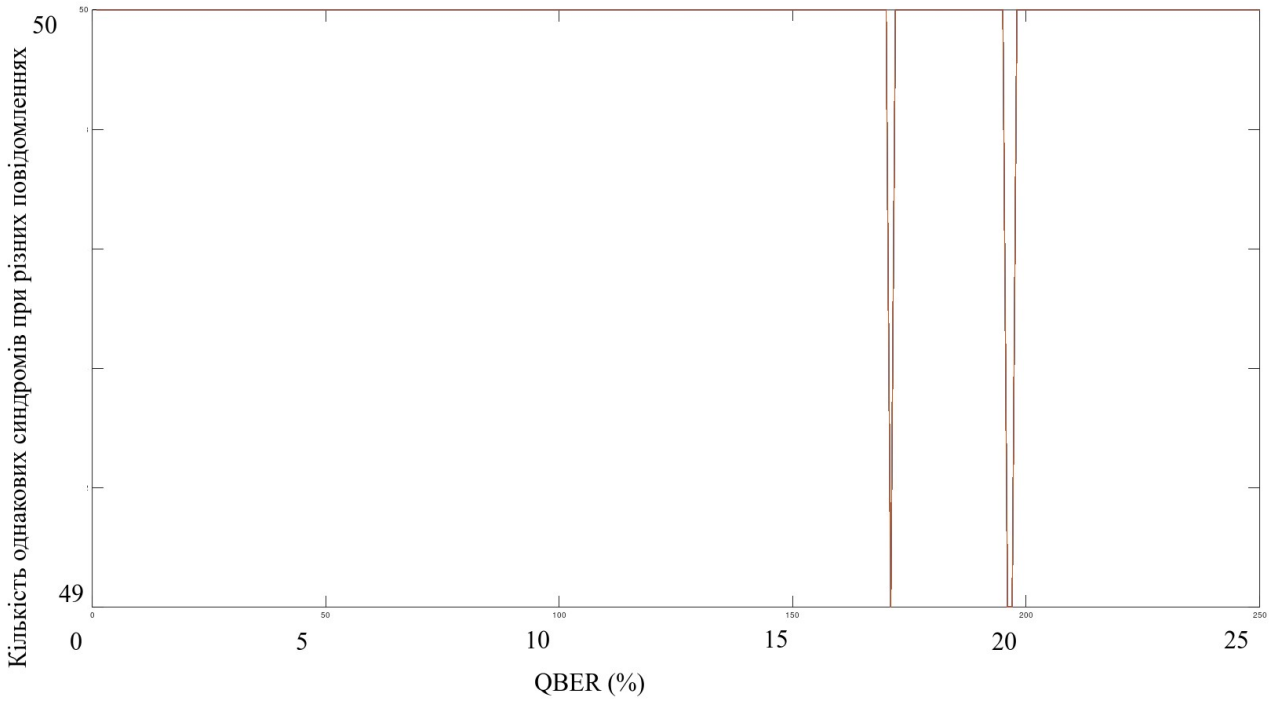


Рисунок 2.7 - Кількість однакових синдромів при різних повідомленнях для різних QBER для матриці з чотирма “1” в кожному стовпці

Як видно з графіку, дана матриця майже не містить однакових синдромів при різних повідомленнях. Лише поодинокі випадки виникнення спільного синдрому при значенні QBER більше 15%. Однак, неможливо повністю позбавитись від спільних синдромів при різних повідомленнях.

Далі проведено дослідження матриці на усі можливі випадки виникнення спільних синдромів. Немає необхідності досліджувати усі варіанти для 1000-бітного просіяного ключа, адже це б зайняло дуже довгий час. Напроти, достатньо дослідити усі можливі варіанти для 20-бітного повідомлення, адже 1000-бітний просіяний ключ буде містити різні комбінації усіх окремих повідомлень. Для цього сформовано випадковим чином повідомлення та

створено синдром для нього. Після цього методом перебору змінювались спочатку один біт в кожній можливій позиції повідомлення, потім два, три і т.д. до двадцяти. Після кожної зміни оригінального повідомлення генерувався новий синдром та порівнювався з синдромом оригінального повідомлення. Зрозуміло, що всього знайшлося 1023 повідомлення, відмінні від оригінального, синдроми яких збігались з синдромом оригінального повідомлення (яке було 1024-тим). Однак, при знаходженні спільних повідомлень також враховувалась кількість біт, які були змінені. В результаті для кожної можливої матриці (з кількістю “1” в кожному стовпці від однієї до п’яти) підраховано кількість спільних синдромів при певній кількості змінених біт та побудовано графіки. Найбільш очікуваним був результат для матриці, в якій кількість “1” в кожному стовпці становила чотири (рис. 2.8.).

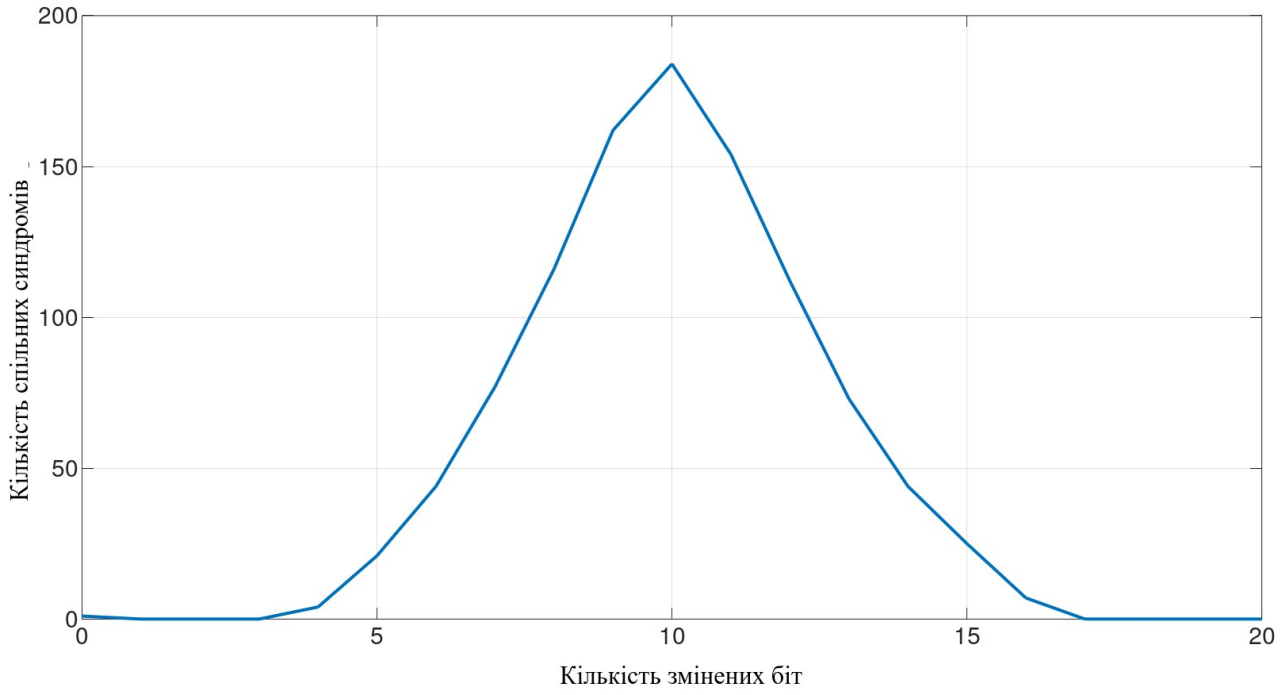


Рисунок 2.8 - Графік залежності числа спільних синдромів
від кількості змінених біт

Цей графік демонструє, що при кількості помилок біт 3 і менше — повідомлення не будуть мати спільних синдромів з оригінальним повідомленням без помилок.

Чому дана матриця є оптимальною? Як визначалось вище, в даний момент в сучасних QKD системах значення QBER в середньому складає 5% (чи навіть менше). Застосовуючи ці умови до даного експерименту це означає, що кількість помилкових біт в середньому складає 50, тобто, по одному помилковому біту в кожній з 50 частин просіяного ключа. Так як в такому повідомленні 20 біт складає 100% всіх бітів, один помилковий біт складатиме 5% помилкових бітів. Два помилкові біти складатимуть 10%, три — 15%, що набагато більше, ніж QBER = 5%. Дана матриця дозволяє генерувати синдроми такі, що маючи в повідомленні 15% помилкових біт в результаті не виявиться спільних синдромів.

Розташування одиниць в матриці залежить від параметру seed, який вводиться вручну оператором при генерації матриці або може бути заздалегідь вказаний в програмі Ніла. Результати на рис. 2.8. відносяться до матриці, значення seed для якої становило 1. Так як даний параметр є 32-бітним, можна вказати будь-яке число в діапазоні від 0 до 65535. Дана кількість варіацій для генерації матриці перевірки є дуже великою. В нашому випадку було досліджено перші 1000 варіантів значення seed. В результаті генерувались матриці, які могли містити спільні синдроми при змінених трьох, двох та одному біті. Але досить часто, в 137 випадках з 1000, генерувались матриці, в яких при змінених 3 бітах і менше не виникало спільних синдромів. Не було знайдено матриці, для якої при змінених 4 бітах не виникало спільних синдромів. Можливо, при інших значеннях seed така матриця може бути виявлена, але у рамках даного дослідження матриця для 3 змінених біт є достатньою.

В результаті рис. 2.8. засвідчує, що знайдено найбільш оптимальну матрицю для реалізації методу Валенти, застосування якої для QBER = 5% з великою долею вірогідності позбавляє виникнення спільних синдромів при різних повідомленнях.

РОЗДІЛ 3. МОДИФІКОВАНИЙ МЕТОД УЗГОДЖЕННЯ КЛЮЧА НА ОСНОВІ LDPC КОДІВ В СИСТЕМАХ QKD ТА ЙОГО ДОСЛІДЖЕННЯ

3.1. Модифікований метод узгодження ключа на основі LDPC кодів в системах QKD

В попередньому розділі визначено найбільш оптимальну матрицю для методу Валенти корекції помилок, яка для QBER = 5% з великою долею вірогідності позбавляє виникнення спільних синдромів при різних повідомленнях. Однак, як і раніше, не вирішеною залишається проблема у наявності великої кількості повідомлень, які просто відкидаються. Нижче запропоновано підходи та рішення, яким чином ці помилки можна виправити.

Перш за все згадаємо, що коди Хемінга гарантовано виправляють помилку в кодовому слові, якщо вона єдина. Як було зазначено в розд. 2, при QBER = 5% можна вважати, що кожне повідомлення поділеного просіяного ключа містить одну помилку. Однак, в реальності кількість помилок може бути дві чи більше.

Запропонована Маккеєм та Нілом процедура декодування використовує не синдроми, а декодує повністю все повідомлення методом найбільшої правдоподібності із застосуванням мережі Байеса. В даному випадку, Боб, маючи лише синдром від Аліси, зрозуміло, застосовувати вище зазначену процедуру декодування не може. Більш того, Боб не може ніяк декодувати або зробити щось з отриманим синдромом Аліси, окрім лише того, щоб провести звіряння його зі своїм. На реалізації цієї ідеї і базується запропонований автором цієї роботи модифікований метод узгодження ключа.

Суть його полягає у реалізації наступної послідовності процедур. Повідомлення Аліси та Боба мають повністю різні синдроми, але їх повідомлення майже однакові і відрізняються лише на декілька біт.

Запропоновано спочатку змінювати по одному біту в кожній позиції та генерувати з нового отриманого повідомлення новий синдром, після чого порівнювати цей синдром з отриманим синдромом від Аліси. Якщо синдроми виявились однакові — знайшли помилковий біт та виправили його. Якщо синдроми виявились різними — змінюємо наступний один біт. Якщо в результаті зміни одного біта в 20 різних позиціях знайти спільний синдром не вдалося — припускаємо, що помилок виявилось дві. Тоді шляхом перебору змінюємо по 2 біта в кожній можливій позиції та створюємо синдроми, після чого порівнюємо їх. Якщо знову не вдалось знайти спільний синдром — повторюється операція, але змінюються вже три біти. Якщо не було знайдено знову спільний синдром — помилок більше ніж 3 і дане повідомлення відкидається. Оскільки в розд. 2 було визначено матрицю перевірки H, яка не містить спільних синдромів при кількості помилкових біт 3 або менше, при наявності в повідомленні 3 або менше помилок шляхом перебору гарантовано після виправлення цих бітів з'явиться синдром, спільний з синдромом Аліси і не буде інших спільних синдромів при інших змінених трьох бітах повідомлення. Такі випадки можуть виникнути, коли кількість помилкових біт 4 або більше через те, що отримана матриця перевірки H вже містить спільні синдроми при різних повідомленнях. Тому, якщо кількість помилок 4 і більше — дане повідомлення відкидається. Однак, слід зазначити, що 4 помилки в повідомленні — це вже QBER = 20%, що набагато вище очікуваних 5%. Тому такі ситуації будуть дуже рідкими. А вірогідність, що з цих повідомлень, які містять 4 помилки, з’явиться спільний синдром, складає близько 0,5%. Перемножуючи ці вірогідності, отримаємо дуже малий відсоток. Для ситуацій, коли повідомлення містить 5 помилок, вірогідність спільного синдрому буде ще нижче. Можна застосовувати дану послідовність процедур по знаходженню спільного синдрому для випадків, коли кількість помилок 4 і більше, але це буде потребувати суттєвого збільшення часу знаходження помилок, тому набагато ефективніше відкинути дане повідомлення, ніж виправляти його.

Однак, можливі також і наступні ситуації: маємо вихідне повідомлення (кількість біт зменшена для демонстрації прикладу), в якому усі біти нульові:

0000000000

В цьому повідомленні з’явились дві помилки і два біти інвертувались, в результаті мається наступне повідомлення:

0011000000

Під час виправлення помилок змінюються кожні два біти, створюються нові синдроми. В результаті при зміні перших двох біт з’явилось наступне повідомлення:

1111000000

Його синдром виявився спільним з синдромом оригінального повідомлення. Однак самі повідомлення є різними. Виникла ситуація, яка не була врахована в попередніх діях.

Пропонується виправляти її наступним чином. Розглянемо декілька можливих ситуацій:

- Наявна 1 помилка в повідомленні. В цьому випадку повідомлення Боба вже відрізняється від повідомлення Аліси на 1 біт, а шляхом перебору та пошуку помилкового біта Боб в 19 випадках створює повідомлення, в якому 2 біти відрізняються від повідомлення Аліси і лише в одному вірному випадку повідомлення не будуть відрізнятися. У випадку, коли відрізняються 2 біти, ситуація є нормальною, синдроми отриманих повідомлень будуть відрізнятися від синдрому Аліси. Тому у цьому випадку метод не потребує удосконалення.
- Наявні 2 помилки у повідомленні. У цьому випадку повідомлення Аліси та Боба можуть не відрізнятись (єдиний вірний випадок), так і відрізнятись на 1, 2, 3 або 4 біти. У перших трьох випадках це не є проблемою, але коли повідомлення відрізняються на 4 біти, є вірогідність згенерувати синдром, спільний з синдромом Аліси, як було показано на прикладі вище. Для цього випадку запропоновано рахувати кількість

спільних синдромів. Якщо при проходженні усіх можливих комбінацій, в яких змінювались 2 біти, знайдено лише один єдиний синдром — цей синдром є синдромом повідомлення, яке є таким ж, як і повідомлення Аліси. Якщо кількість спільних синдромів виявилась більше 1 — це повідомлення просто відкидається.

- Наявні 3 помилки у повідомленні. У цьому випадку, як і у випадку з 2 помилками, повідомлення Аліси та Боба можуть не відрізнятись (єдиний вірний випадок), так і відрізнятись на 1, 2, 3, 4, 5 або 6 бітів. У перших трьох випадках це не є проблемою, але коли повідомлення відрізняються на 4, 5, 6 бітів є вірогідність згенерувати синдром, спільний з синдромом Аліси, як було показано на прикладі вище. Для цього випадку також запропоновано рахувати кількість спільних синдромів. Якщо при проходженні усіх можливих комбінацій, в яких змінювались 3 біти, знайдено лише один єдиний синдром — цей синдром є синдромом повідомлення, яке є таким ж, як і повідомлення Аліси. Якщо кількість спільних синдромів виявилась більше 1 — це повідомлення просто відкидається.
- Наявні 4 помилки у повідомленні. У цьому випадку ще не відомо, що помилок 4. Продемонструємо це наступним чином:

  0000000000

  оригінальне повідомлення містить усі нулі.

  1111000000

  Повідомлення з помилками містить 4 помилки. Розглянемо далі 5 випадків зміни одного біта:

  0111000000

  1011000000

  1101000000

  1110000000

  1111100000

В перших чотирьох випадках повідомлення відрізняються на 3 біти, отже спільних синдромів виникнути не повинно. Однак в останньому випадку повідомлення відрізняються на 5 біт, можливе виникнення спільного синдрому, але Боб вважає, що змінився лише один біт. Саме для таких випадків, коли відбувається перевірка лише на один біт, пропонується наступний підхід: замість зміни в кожній позиції одного біту та генерації нового синдрому, Боб бере свій синдром та синдром Аліси та проводить логічну операцію XOR. При наявності однієї помилки даний вектор гарантовано збігатиметься з одним стовпцем матриці перевірки. Номер стовпця, який збігся з отриманим вектором буде вказувати на номер біта, в якому виявилась помилка. Якщо ніякий стовпець не збігся з отриманим вектором, це означає, що в повідомленні більше ніж одна помилка. Даний метод по швидкості роботи є дещо швидшим за запропонований вище метод перевірки одного біта, але позбавляє поточний випадок, коли помилок 4, створити помилково спільний синдром з оригінальним повідомленням. Розглянемо далі наступні ситуації:

0011000000

0101000000

...

0111100000

0111010000

...

1111110000

1111101000

...

1111000011

В таких випадках, коли змінюються 2 біти, можлива ситуація, коли виникає низка повідомлень, в яких відрізняються 4 або 6 бітів в

повідомленнях. І якщо у випадку з 6 зміненими бітами співвідношення комбінацій, при яких генерується спільний синдром до усіх можливих комбінацій, є дуже малим (за рахунок загальної кількості можливих комбінацій, рис. 3.1), у випадку з 4 зміненими повідомленнями співвідношення комбінацій, при яких генерується спільний синдром до усіх можливих комбінацій, приблизно в 4 рази більше.

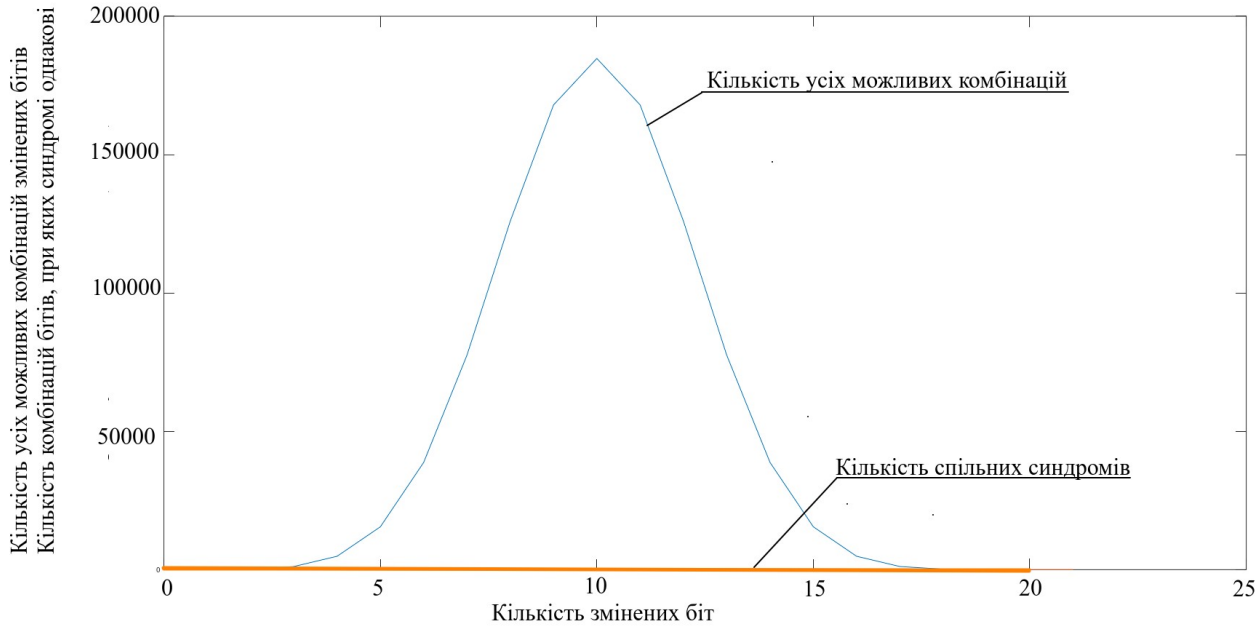


Рисунок 3.1 - Графік залежності усіх можливих комбінацій змінених бітів (синій колір) та окремо комбінацій для бітів, при яких синдроми однакові (оранжевий колір) від кількості змінених біт

Зупинимось більш детальніше рис. 3.1. На даному графіку синім кольором показано кількість можливих комбінацій (комбінаторика) з певної кількості біт, які змінюються відносно оригінального повідомлення. Тобто у випадку, якщо змінюється лише 1 біт — можливих комбінацій 20, якщо 2 — 190 і т.д. Оранжевим кольором відображається кількість спільних синдромів при певній кількості змінених біт (рис. 2.8.) Однак, вона є мізерно малою, тому

на цьому рисунку нас цікавить синій графік і чим він нижче для певної кількості біт — тим вище вірогідність створити спільний синдром.

Такий же самий підхід застосовується у випадку, якщо 4 біти є помилковими, коли йде перебір і змінюються 3 біти.

Такий підхід справедливий і для випадків, коли кількість змінених біт 5 та більше. Але знову таки, 4 змінені біти складають 20% помилок, а більше число змінених бітів складає ще більший відсоток. Тим не менше наразі найслабкішим місцем запропонованого методу є саме випадок, коли в реальності змінено 4 біти і при зміні 2 або 3 біт також виникають 4 помилки.

Насамкінець, слід зазначити, що і цю вірогідність можна зменшити, якщо знайти матрицю, яка у випадку зміни 4 біт матиме мінімальну кількість спільних синдромів, а саме – лише один. Таку матрицю було визначено під час генерації матриці з певним параметром seed.

3.2. Алгоритм реалізації модифікованого методу

В результаті зазначена вище послідовність процедур може бути представлена у вигляді наступного алгоритму виправлення помилок:

1. Аліса та Боб ділять свої просіяні ключі на 50 частин.
2. Для кожної частини (повідомлення) генеруються синдроми.
3. Аліса відправляє синдром до Боба.
4. Боб порівнює синдроми, якщо вони збіглись — зберігає дане повідомлення.
5. Якщо синдроми різні, Боб проводить між синдромами логічну операцію XOR, та результатний вектор порівнює з кожним стовпцем матриці перевірки, вважаючи, що в повідомлені лише одна помилка.

6. Якщо знайшовся стовпець, однаковий з результатним вектором — виправляється той біт, номер якого є номером стовпця, який збігся з результатним вектором.
7. Якщо вектор не дорівнює жодному стовпцю, Боб перевіряє своє повідомлення, змінюючи два біта в кожних можливих позиціях, після чого створює синдром та порівнює його з синдромом Аліси.
8. Якщо в результати знайшовся один єдиний вектор, цей вектор відповідає повідомленню, яке є спільним з оригінальним повідомленням; якщо синдромів виявилось два чи більше — отже в повідомленні не дві помилки а набагато більше, і дане повідомлення відкидається.
9. Якщо не знайдено жодного синдрому, Боб перевіряє своє повідомлення, змінюючи три біта в кожних можливих позиціях, після чого створює синдром та порівнює його з синдромом Аліси.
10. Якщо в результати знайшовся один єдиний вектор, цей вектор відповідає повідомленню, яке є спільним з оригінальним повідомленням; якщо синдромів виявилось два чи більше — отже в повідомленні не дві помилки а набагато більше, і дане повідомлення відкидається.
11. Якщо не знайдено жодного синдрому, Боб відкидає дане повідомлення.
12. Боб надсилає результати перевірки Алісі, яка відкидає свої повідомлення у відповідності до відкинутих Бобом.

На рис 3.2 знаходиться схема запропонованого методу корекції помилок.

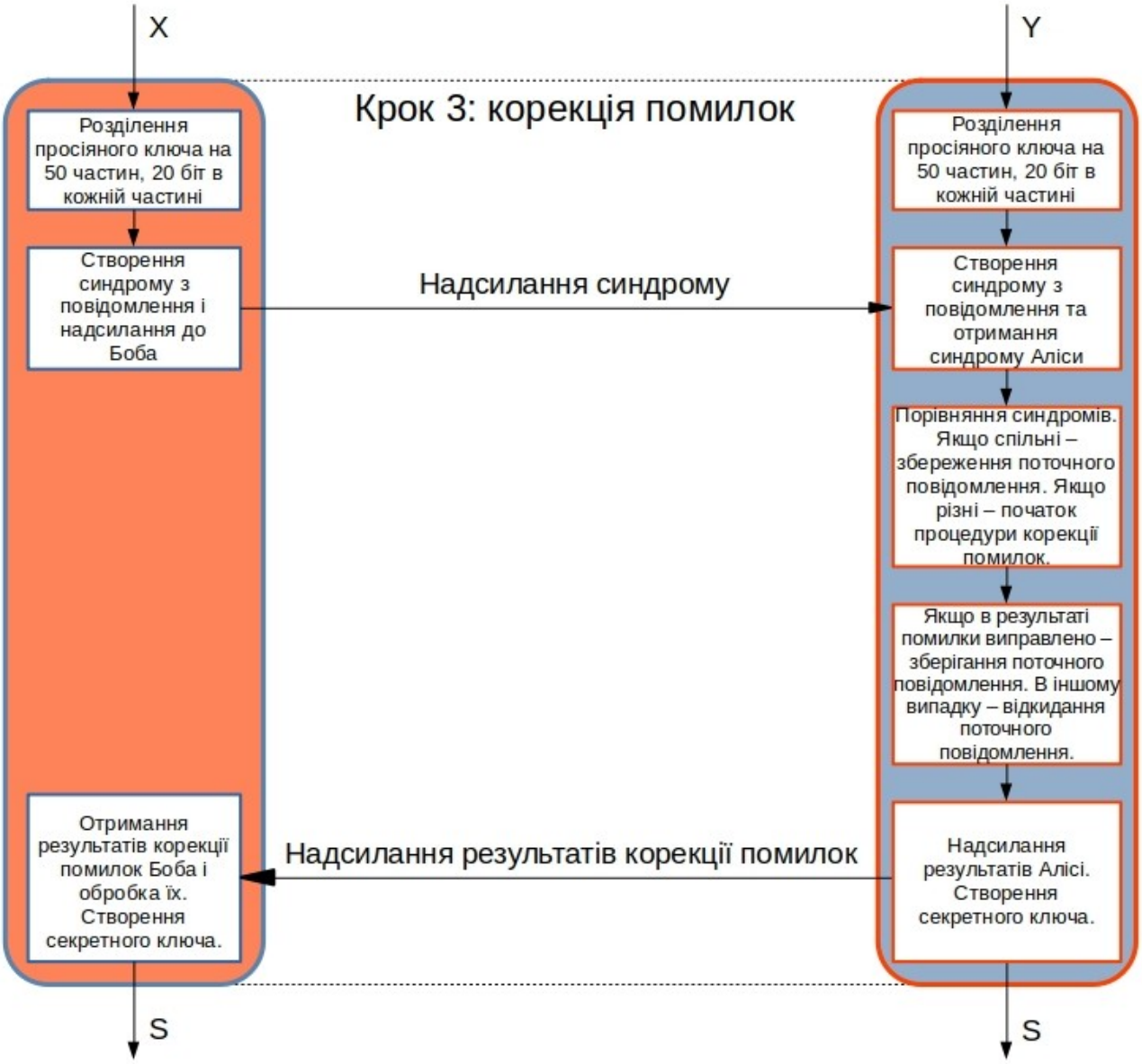


Рисунок 3.2 - Модифікований алгоритм корекції помилок

3.3. Моделювання процедури корекції помилок модифікованим методом

Для дослідження запропонованого методу корекції створено програму на мові С (лістинг наведено в Додатку Г), яка функціонує наступним чином:

1. Просіяні ключі діляться на 50 частин та створюються синдроми. Лічильник дорівнює нулю.

2. Порівнюються синдроми, якщо синдроми спільні, лічильник +1.
3. Якщо синдромі не спільні, перевіряються описаним вище алгоритмом випадки для 1 2 чи 3 помилкових біт. Якщо в результаті вдалось виправити помилки — лічильник +1.
4. У всіх інших випадках кількість помилок 4 чи більше, лічильник +0.

Очевидно, що максимальне число, яке може міститись в лічильнику, є 50.

Для більш повного представлення картини діапазон змінювання значення QBER склав від 0,1% до 25% з кроком 0,1%. При цьому для кожного значення QBER експеримент повторено 100 разів, а з отриманих результатів обчислено середнє арифметичне значення. Результат наведено на рис. 3.3.

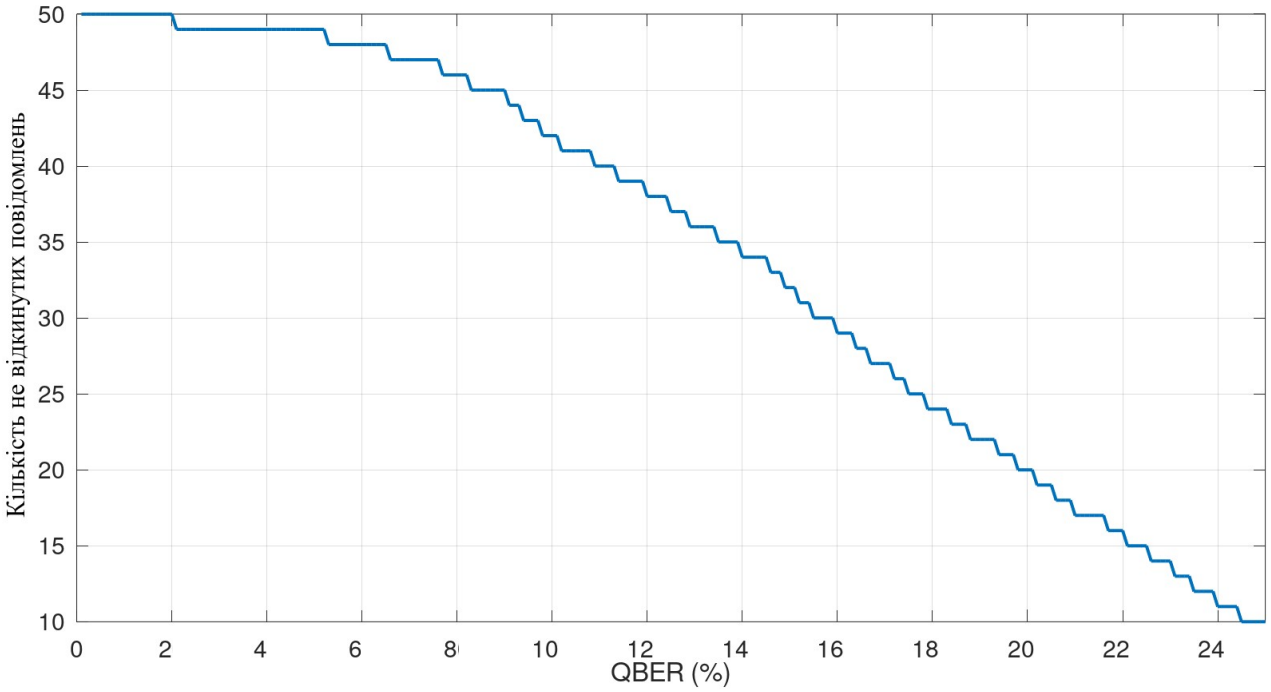


Рисунок 3.3 - Усереднена кількість спільних повідомлень для різних QBER згідно модифікованого методу корекції помилок

Порівнюючи отримані результати (рис. 3.3) з результатами моделювання процедури виправлення помилок методом Валенти (рис. 2.4) слід зробити висновок про те, що реалізація модифікованого методу корекції дозволяє виправляти майже всі помилки та дає змогу зберегти достатню довжину просіяного ключа, який є виправленим та може переходити на наступний етап (Step 4, рис. 2.1) в якості секретного ключа.

РОЗДІЛ 4. РОЗРОБКА СТАРТАП-ПРОЕКТУ

4.1. Опис ідеї проекту

Запропоновано використання *H*-матриці корекції, яка не буде створювати однакові синдроми повідомлень до трьох помилок, що дає змогу запропонувати модифікований метод узгодження ключа шляхом виявлення та корекції помилок на основі LDPC кодів в QKD системах, який відрізняється від відомого методу (запропонованого Валентою) введенням процедури виправлення помилкових повідомлень за рахунок перебору можливих варіантів нових повідомлень та порівняння їх синдромів з синдромами повідомлення передавальної сторони, що дає змогу суттєво підвищити надійність створюваної QKD системи.

Далі послідовно проаналізовано та подано у вигляді таблиць: зміст ідеї; можливі напрямки застосування; основні переваги, які може отримати користувач товару та чим відрізняється від існуючих аналогів та замінників.

Таблиця 4.1 - Опис ідеї стартап-проекту

Зміст ідеї	*Напрямки застосування*	*Переваги для користувача*
	1. Корекція помилок в системах QKD	Корекція до трьох включно помилок в кожному повідомлені просіяного ключа.
	2. Збереження конфіденційності в системах QKD.	Забезпечення конфіденційності просіяного ключа на етапі корекції помилок від третіх сторін.

Висновки: в табл. 4.1 наведено основні напрямки використання запропонованого рішення. Споживачами даної продукції можуть бути як компанії для застосування в телекомунікаційних системах, так і державні установи для побудови захищених систем.

Таблиця 4.2 - Визначення сильних, слабких та нейтральних характеристик

№ п/п	*Техніко-економічні характеристики ідеї*	*(потенційні) товари/концепції конкурентів*				*W*	*N*	*S*
		Мій проект	Конку-рент1	Конку-рент2	Конку-рент3			
1.	Собівартість	10	15	15	20			+
2.	Продуктивність	13250000	10000000	12000000	13000000			+
3.	Розмір	15*10	10*10	8*8	7*8	-		
4.	Інтерфейс зв’язку	Ethernet	Ethernet	Ethernet	Ethernet		+	
5.	Масштабованість	є	є	є	є		+	
6.	Споживання	1	1	1	0,8	-		

В табл. 4.2 W – слабка сторона, N – нейтральна сторона, S – сильна сторона. Під масштабованістю розуміється можливість зменшення розмірів системи в майбутньому.

Висновки: у порівнянні з конкурентами товар має перевагу у кращому відношенні ціна/продуктивність. Масштабованість притаманна усім системам. Система споживає стільки же енергії, скільки і інші типові представники. Так як

система призначена для телекомунікаційних систем, наразі єдиним інтерфейсом зв’язку, для яких розробляються системи, є Ethernet.

4.2. Технологічний аудит ідеї проекту

Таблиця 4.3 - Технологічна здійсненність ідеї проекту

№ п/п	*Ідея проекту*	*Технології її реалізації*	*Наявність технологій*	*Доступність технологій*
1.	Підвищення продуктивності обчислення	Використання паралельного обчислення шляхом розділення перевірки синдромів між двома сторонами.	Наявна	Доступна
2.	Використання FPGA платформи	Використання розробленого апаратно-програмного комплексу	Наявна	Доступна
Обрана технологія реалізації ідеї проекту: за основу можна поєднати два пункти, і їх використання дозволить продукту більше виділятись на ринку відносно конкурентів.				

4.3. Аналіз ринкових можливостей запуску стартап-проекту

Таблиця 4.4 - Попередня характеристика потенційного ринку стартап-проекту

№ п/п	*Показники стану ринку (найменування)*	*Характеристика*
1.	Кількість головних гравців, од	3
2.	Загальний обсяг продаж, грн/ум.од	70000
3.	Динаміка ринку (якісна оцінка)	зростає
4.	Наявність обмежень для входу (вказати характер обмежень)	немає
5.	Специфічні вимоги до стандартизації та сертифікації	відсутні
6.	Середня норма рентабельності в галузі (або по ринку), %	45%

Висновки: проаналізувавши табл. 4.4 можна зазначити, що вихід на ринок є рентабельним, так як мала кількість гравців на ринку, що свідчить про низьку конкуренцію та високий відсоток рентабельності, що дає змогу швидко покрити витрачені кошти на розробку системи.

Таблиця 4.5 - Характеристика потенційних клієнтів стартап-проекту

№ п/п	*Потреба, що формує ринок*	*Цільова аудиторія (цільові сегменти ринку)*	*Відмінності у поведінці різних потенційних цільових груп клієнтів*	*Вимоги споживачів до товару*
1.	Збільшення продуктивності та надійності сучасних телекомунікаційних мереж	Державний сектор, приватний сектор	Інтеграція з існуючими системами, необхідність швидких та надійних телекомунікаційних мереж	Продуктивність, енерго-споживання, висока надійність

Висновки: формування ринку визначається потребою збільшення продуктивності та надійності телекомунікаційних мереж. Основними споживачами продукту є усі сфери, які прагнуть збільшити автоматизацію процесів, що використовуються. Тому головними вимогами до товару є продуктивність та надійність роботи.

Таблиця 4.6 - Фактори загрози

№ п/п	*Фактор*	*Зміст загрози*	*Можлива реакція компанії*
1.	Економічний	Економічний стан країни-виробника	Зміна країни-виробника
2.	Якісний	Не належна якість зібраного приладу	Зміна технологічних процесів виробництва
3.	Вартість комплектуючих	Підвищення закупівельної вартості комплектуючих	Пошук нових постачальників
4.	Конкуренція	Ім’я конкурентів є більше відомим на ринку	Проведення потужної рекламної кампанії
5.	Політичний	Політична ситуація країни-виробника	Зміна країни виробника

Висновки: основними факторами загрози є конкуренція та економічно-політичний стан країни виробника. Існуючі товари вже мають певне ім’я, репутацію та об’єми виробництва. Також економічна та політична ситуація країни-виробника може зіграти значну роль у втраті прибутку.

Таблиця 4.7 - Фактори можливостей

№ п/п	*Фактор*	*Зміст можливості*	*Можлива реакція компанії*
1.	Збільшення попиту	Різке збільшення зацікавленості до продукту	Підвищення виробництва
2.	Новітні технології	Можливість розгортання повномасштабних телекомунікаційних мереж у повсякденному	Співпраця з іншими компаніями в даній сфері
3.	Розширення кругозору компанії	Можливість додавання нових систем до існуючої для пришвидшення розвитку телекомунікаційних мережі вцілому	Відкриття нових спеціалізованих підрозділів компанії
4.	Індивідуальне замовлення	Можливість додавати індивідуальні потреби для клієнтів	Проведення аналізу раціональності замовлення та можливість укладання нового контракту із заданими потребами
5.	Кооперація із лідерами ринку	Конкуренти запропонували об’єднання компаній	Оцінка можливих переваг та ризиків об’єднання

Висновки: сфера ринку квантових телекомунікаційних мереж є відносно новою та швидко розвиваючою. Квантові технології вводяться всюди, що спричиняє зростання клієнтів на ринку, які в свою чергу збільшують попит на запропоновану систему в тому числі. Це приведе до збільшення об'ємів виробництва та заключення великої кількості контрактів, що в свою чергу створює вигідні економічні можливості для дослідження нових технологій та покращення існуючої системи.

Таблиця 4.8 - Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	*В чому проявляється дана характеристика*	*Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)*
Вказати тип конкуренції - чиста	Мала кількість постачальників даного товару	Розвивати систему збільшуючи її продуктивність та надійність
За рівнем конкурентної боротьби - міжнародний	Наявність замовників та виробників із інших держав	Вихід на міжнародний ринок
3. За галузевою ознакою - міжгалузева	Використання у різних галузях	Проведення потужної рекламної кампанії
4. Конкуренція за видами товарів – товарно-видова	Запропонований товар є одного виду	Орієнтація стратегії компанії на клієнта та адаптація до змін ринкових умов
5. За характеристиками конкурентних переваг - неціновa	Основним є якість та надійність товару	Проведення робіт щодо постійного покращення продукту
6. За інтенсивністю - марочна	Бренд грає велику роль в постачанні продукту	Проведення рекламної кампанії та доведення якості продукту

Висновки: ринок є конкурентним, проте вид конкуренції є чистим, так як окремі гравці мало впливають на ціну товару. Конкурентний ринок є міжнародним та міжгалузевим. Конкуренція за видами товарів – видова.

Таблиця 4.9 - Аналіз конкуренції в галузі за М. Портером

Складові аналізу	*Прямі конкуренти в галузі*	*Потенційні конкуренти*	*Постачальники*	*Клієнти*	*Товари-замінники*
	SK Telecom Samsung IBM Cisco	Intel KT	Thor	Держ. та приватний сектори	Відеокарти, FPGA
Висновки:	Конкуренція є низькою	Вихід на ринок є відносно простим. Наявні потенційні конкуренти	Постачальники не мають диктувати ціни на ринку	Клієнти можуть диктувати умови через присутність компаній з хорошою репутацією	Існують обмеження по використанню

Таблиця 4.10 - Обґрунтування факторів конкурентоспроможності

№ п/п	*Фактор конкурентоспроможності*	*Обґрунтування (наведення чинників, що роблять фактор для порівняння)*
1.	Собівартість	Низька собівартість – більша доступність кінцевого пристрою
2.	Продуктивність	Більша продуктивність в порівнянні з конкурентами
3.	Надійність	Збільшення продуктивності квантової мережі збільшує її надійність

Висновки: підвищення продуктивності завдяки введення процедури корекції помилок в повідомленнях є доволі сильною стороною системи, що в свою чергу збільшує надійність квантової мережі, що є також її сильною стороною. Також низька собівартість робить систему більш конкурентоспроможною.

Таблиця 4.11 - Порівняльний аналіз сильних та слабких сторін проекту

№ п/п	*Фактор конкурентоспроможності*	*Бали 1-20*	*Рейтинг товарів-конкурентів у порівнянні з SK Telecomunication*						
			-3	*-2*	*-1*	*0*	*1*	*2*	*3*
1.	Собівартість	15							+3
2.	Продуктивність	20		-2					
3.	Надійність	19			-1				

Висновки: аналізуючи табл. 4.11 можна зробити висновок, що запропонована система має більший рейтинг відносно головного конкурента. Дана таблиця демонструє основні особливості продукту, які відрізняють його від основного конкурента.

Таблиця 4.12 - SWOT-аналіз стартап-проекту

Сильні сторона: Низька собівартість Висока надійність	Слабкі сторони: Відносно високе енергоспоживання Не досить компактний корпус
Можливості: Вихід на міжнародний ринок Збільшення попиту	Загрози: Конкуренція Економічна нестабільність Політична нестабільність

Таблиця 4.13 - Альтернативи ринкового впровадження стартап-проекту

№ п/п	*Альтернатива (орієнтований комплекс заходів) ринкової поведінки*	*Ймовірність отримання ресурсів*	*Строки реалізації*
1.	Максимізація власного виграшу (індивідуалізм)	Середня	15 місяців
2.	Максимізація спільного виграшу (кооперація)	Висока	18 місяців
3.	Суперництво	Середня	24 місяці

Висновки: було обрано кооперацію як альтернативну ринкову поведінку, так як за відносно не високий термін існує велика ймовірність отримання ресурсів.

4.4. Розроблення ринкової стратегії проекту

Таблиця 4.14 - Вибір цільових груп потенційних споживачів

№ п/п	*Опис профілю цільової групи потенційних клієнтів*	*Готовність споживачів сприйняти продукт*	*Орієнтовний попит в межах цільової групи (сегменту)*	*Інтенсивність конкуренції в сегменті*	*Простота входу у сегмент*
1.	Державний сектор	-	+	висока	-
2.	Приватний сектор	+	+	висока	+

Які цільові групи обрано: основною характеристикою вибору цільової групи є готовність прийняти продукт. В даній області приватний сектор є більш готовим, адже державний сектор потребує більше дозволів та роз'яснень для введення нового продукту в системи.

Таблиця 4.15 - Визначення базової стратегії розвитку

№ п/п	*Обрана альтернатива розвитку проекту*	*Стратегія охоплення ринку*	*Ключові конкурентос-проможні позиції до обраної альтернативи*	*Базова стратегія розвитку*
1.	Індивідуалізм	Стратегія недиференці-йованого маркетингу	Адаптація до вимог ринку Використання новацій	Стратегія спеціалізації

Висновки: через існування на ринку більш сильних та розкручених гравців було обрано стратегію розвитку спеціалізація.

Таблиця 4.16 — Стартові умови проекту

№ п/п	*Чи є проект «першопрохідцем» на ринку?*	*Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?*	*Чи буде компанія копіювати основні характеристики товару конкурента, і які?*	*Стратегія конку-рентної поведінки*
1.	Не є першопрохідцем на ринку	Буде як шукати нових споживачів, так і забирати вже існуючих	Компанія не буде копіювати основні характеристики конкурента	Стратегія виклику лідера

Висновки: оскільки на ринку вже є проекти-конкуренти, компанія може обрати стратегію виклику лідера, так як проект має переваги. Також можлива колаборація з конкурентами для досягнення кращого успіху, адже система є новою та ще тільки вивчається та досліджується. Можливість об’єднати зусилля дає змогу в майбутньому краще засвоїти це направлення та створювати кращі системи.

Таблиця 4.17 - Визначення стратегії позиціонування

№ п/п	*Вимоги до товару цільової аудиторії*	*Базова стратегія розвитку*	*Ключові конкурентоспроможні позиції власного стартап-проекту*	*Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)*
1.	Продуктивність	Стратегія спеціалізації	Продуктивна	Висока швидкодія роботи
2.	Надійність	Стратегія спаціалізації	Якість	Висока надійність роботи

Висновки: як зазначалось раніше, збільшення продуктивності збільшує і надійність системи, що повинно викликати довіру до продукту у споживачів.

4.5. Розроблення маркетингової програми стартап-проекту

Таблиця 4.18 - Визначення ключових переваг концепції потенційного товару

№ п/п	*Потреба*	*Вигода, яку пропонує товар*	*Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)*
1.	Введення швидкої та надійної системи комунікації в квантовій мережі	Висока продуктивність та надійність	Ціна, продуктивність, надійність

Висновки: визначившись з основними перевагами концепції товару, можливе створення відповідної рекламної кампанії для кінцевих клієнтів.

Таблиця 4.19 - Опис трьох рівнів моделі товару

Рівні товару	*Сутність та складові*		
I. Товар за задумом	Пристрій дає змогу втілити повноцінні квантові мережі в реальне життя за рахунок швидкої та надійної системи комунікації між квантовими девайсами.		
II. Товар у реальному виконанні	Властивості/характеристики Швидкодія Вартість	М/Нм 13250000 hash/s 10	Вр/Тх /Тл/Е/Ор Тх Тх
	Якість: стандарт вологозахисту IPx8		
	Пакування: система, блок живлення, документація користувача та гарантійний талон		
	Марка: назва організації-розробника – Quantum development group, назва товару – Quantum transceiver		
III. Товар із підкріпленням	До продажу – комплектація, яку вимагає замовник, установка.		
	Після продажу – сервіс, гарантій, сервісне обслуговування		
Товар захищатиметься шляхом його патентування			

Висновки: шляхом патентування товару створюється захист від його копіювання. Також закладені характеристики на другому рівні товару роблять його досить унікальним та конкурентоспроможним.

Таблиця 4.20 - Визначення меж встановлення ціни

№ п/п	*Рівень цін на товари замінники*	*Рівень цін на товари-аналоги*	*Рівень доходів цільової групи споживачів*	*Верхня та нижня межі встановлення ціни на товар/послугу*
1.	500-1500 у.о	400-2000 у.о	1500-5000 у.о.	500-2000 у.о.

Висновки: обрано середню категорію цін, адже занадто велика ціна відлякує споживачів, проте занадто низька ціна може навести на думку, що товар не є належної якості.

Таблиця 4.21 - Формування системи збуту

№ п/п	*Специфіка закупівельної поведінки цільових клієнтів*	*Функції збуту, які має виконувати постачальник товару*	*Глибина каналу збуту*	*Оптимальна система збуту*
1.	Продаж	Повний супровід товару до замовника	Нульового рівня	Безпосередній (прямий)

Висновки: основним каналом збуту є продаж товару. На старті компанії очікуються відносно невеликі об’єми виробництва, тому на даному етапі можливо обійтись без посередників і продавати товар напряму клієнтам. Саме тому було обрано нульовий рівень глибини каналу збуту та пряму систему збуту.

Таблиця 4.22 - Концепція маркетингових комунікацій

№ п/п	*Специфіка поведінки цільових клієнтів*	*Канали комуні-кацій, якими користуються цільові клієнти*	*Ключові позиції, обрані для позиціону-вання*	*Завдання рекламного повідомленн я*	*Концепція рекламного звернення*
1.	Розвиток технологій спонукає споживача до оновлення власних використо-вуваних систем	Реклама SMM Відео-інструкції по використанню товару на ресурсі youtube.com	Висока продукти-вність та надійність Легкість у встановленні та використанні	Донести можливість полегшення повсяк-денного життя та виробничих процесів	Демон-страція можли-востей даної системи та принцип її викори-стання

Висновки: маркетингова кампанія відбувається за рахунок соціальних мереж та цільових рекламних кампаніях. Метою даних оголошень є донести усі перспективи та можливості даної системи для користувача.

4.6. Можливі області застосування та очікуваний ефект

Даний проект буде застосовуватись та удосконалюватись в Центрі квантової інформації (*Center for Quantum Information*) Корейського інституту науки та технологій (*Korea Institute of Science and Technology, KIST*). Очікуваним ефектом є зниження кількості невиправлених помилок в системі QKD, що

надасть змогу використовувати її замість існуючих криптографічних систем, які потребують більших обчислювальних ресурсів та споживаної енергії.

Позитивний економічний ефект досягається за рахунок меншої собівартості системи в порівнянні з конкурентами.

Висновки

Розроблено перший етап створення стартап-проекту. Оскільки кожна наукова робота повинна знаходити своє місце в застосуванні у реальному житті, тому стартап-проект може бути практичним відображенням наукової праці.

Висвітлено зміст ідеї проекту шляхом розгляду потенційних зацікавлених осіб, які в майбутньому можуть стати клієнтами запропонованої продукції. Також розглянуто ризики реалізації продукції, а проведений аналіз сильних та слабких сторін надав можливість визначити аспекти, на які слід зробити ставку.

Проведено технічний аудит проекту та визначено технології, які використовуватимуться. Запропоновані технології вже існують, проте їх використання не дає можливість переваги над конкурентом.

Проведено аналіз усіх аспектів ринку, який показав, що імплементація проекту можлива в реальних умовах, проте слід враховувати, що на ринку уже існують гравці з досить високою репутацією, що може зіграти негативну роль у впровадженні проекту. Для уникнення провалу проекту необхідно провести потужну рекламну кампанію, в якій донести до споживача усі переваги даного проекту та необхідність обрати саме запропонований продукт.

Загальні висновки

В дисертаційній роботі вирішено актуальну та важливу науково-прикладну задачу підвищення надійності та захищеності систем QKD шляхом подальшого розвитку методу узгодження ключа із застосуванням корекції помилок на основі LDPC-кодів та розроблення алгоритмічних і програмних рішень його реалізації. Під час виконання дослідження отримано наступні наукові та практичні результати:

1. Проаналізовано існуючі методи корекції помилок у системах QKD, а саме Cascade, Winnow і LDPC та обґрунтовано доцільність використання в дослідженнях LDPC-кодів, здатних виправляти більше, ніж одну помилку на блок. Окрім цього, реалізація методів Cascade та Winnow потребує занадто багато часу для виконання повторних ітерацій. Оскільки обмін бітами відбувається постійно, необхідно враховувати відстань між сторонами та довжину кодових слів, а для великих відстаней та довгих повідомлень застосування LDPC-кодів є більш ефективним.

2. Проведено порівняльний аналіз існуючих підходів застосування LDPC-кодів в системах QKD на прикладі методів Валенти та Міліцевіча виправлення помилок. Обґрунтовано вибір в якості базового для подальшого удосконалення метод Валенти, який на відміну від метода Міліцевича використовується в DV-QKD системах, застосовує лише матрицю перевірки та звичайну операцію порівняння, тоді як метод Міліцевича відповідно використовується лише в CV-QKD системах та має бути адаптований до DV-QKD систем, потребує використання окрім матриці перевірки також породжувальної матриці та складних математичних обчислень.

3. Удосконалено метод узгодження ключа в системах QKD шляхом виявлення та корекції помилок на основі LDPC-кодів, який відрізняється від відомого методу Валенти введенням процедури виправлення помилкових

повідомлень за рахунок перебору можливих варіантів нових повідомлень та порівняння їх синдромів з синдромом повідомлення передавальної сторони, що дозволило підвищити надійність систем QKD.

4. Запропоновано використання LPDC *H*-матриці перевірки, яка не буде створювати однакові синдроми повідомлень до трьох помилок, що дало змогу підвищити надійність систем QKD.

5. Розроблено алгоритмічні та програмні рішення реалізації запропонованого модифікованого методу узгодження ключа в QKD системах. Створено програмний комплекс на мові програмування C та проведено моделювання результатів в системі GNU Octave, які підтвердили підвищення надійності QKD систем згідно запропонованих рішень у порівнянні з базовим методом Валенти. Заплановано впровадження одержаних результатів в Центрі квантової інформації (*Center for Quantum Information*) Корейського інституту науки та технологій (*Korea Institute of Science and Technology, KIST*).

6. Розроблено та проаналізовано стартап-проект, який встановив доцільність комерціалізації проекту з використанням модифікованого методу узгодження ключа на основі LDPC-протоколу в системах квантового розповсюдження ключів.

Наукові та практичні результати дисертаційної роботи доповідались і обговорювались на X Міжнародній науково-практичній інтернет-конференції «Сучасний рух науки», м. Дніпро, квітень, 2020р.

За матеріалами дисертації опубліковано 1 друковану працю в збірнику матеріалів конференції (див. Додаток А):

- Bilash Bohdan. The Implementation of the Modified Error Correction Method in Quantum Key Distribution // Збірник тез доповідей X Міжнародної науково-практичної інтернет-конференції «Сучасний рух науки», м. Дніпро, квітень, 2020 р.– С. 109-112.

За матеріалами досліджень також підготовлено та подано до друку у фаховому виданні України 1 статтю (реєстр. № 201253, див. Додаток Б), яка наразі проходить процедуру наукового рецензування:

- Bilash B.O. Optimal low density parity check matrices to correct quantum key errors for QKD // Мікросистеми, Електроніка та Акустика. – 2020.

Вищевикладене дозволяє зробити висновок, що завдання на магістерську дисертацію виконано у повному обсязі, а її мета досягнута.

Список використаних джерел

[1] Chris Savarese and Brian Hart, “The Caesar Cipher, [Електронний ресурс] - режим доступу: www.cs.trincoll.edu/~crypto/historical/caesar.html,” 1999.

[2] A. S. L. A. R. Rivest, “CRYPTOGRAPHIC COMMUNICATIONS SYSTEM AND METHOD Patent 4405829,” vol. 19, №. 54, 1985.

[3] S. Wiesner, “Conveyed coding,” *ACM SIGACT News*, vol. 15, №. 1, P.P. 78–88, 1983, DOI: 10.1145/1008908.1008920.

[4] Simon Singh, “The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography, [Електронний ресурс] - режим доступу: https://books.google.co.kr/books?id=fbp9V9dkaNkC&redir_esc=y,” 2011.

[5] C. Museum, “History of the Enigma, [Електронний ресурс] - режим доступу: https://www.cryptomuseum.com/crypto/enigma/hist.htm.”

[6] C. E. Shannon, “Communication theory of secrecy systems. 1945.,” *MD. Comput.*, vol. 15, №. 1, P.P. 57–64, 1998.

[7] D. W. Leung, “Quantum vernam cipher,” *Quantum Inf. Comput.*, vol. 2, №. 1, P.P. 14–34, 2002.

[8] W. Peng, D. Cheng, and C. Song, “One-time-pad cryptography scheme based on a three-dimensional DNA self-assembly pyramid structure,” *PLoS One*, vol. 13, №. 11, P.P. 1–24, 2018, DOI: 10.1371/journal.pone.0206612.

[9] T. Elgamal, “A public key cryptosystem and a signature scheme based on discrete logarithms,” *IEEE Trans. Inf. Theory*, vol. 31, №. 4, P.P. 469–472, Jul. 1985, DOI: 10.1109/TIT.1985.1057074.

[10] M. Rabin, “Digitalized Signatures and Public-Key Functions as Intractable as Factorization,” *Foundations of Secure Computations*. P.P. 155–168, 1978, DOI: 10.1080/09720529.2013.858478.

[11] Certicom Research, “Standards for efficient cryptography, SEC 1: Elliptic

Curve Cryptography," *Stand. Effic. Cryptogr.*, vol. 1, №. Sec 1, P.P. 1–22, 2009, DOI: 10.1002/smj.

[12] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, 1996, vol. 59, №. 3, P.P. 124–134, DOI: 10.1109/SFCS.1994.365700.

[13] L. M. K. Vandersypen, M. Breyta, G. Steffen, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang, "Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance," *Nature*, vol. 414, №. 6866, P.P. 883–887, 2001, DOI: 10.1038/414883a.

[14] E. Schrödinger, "Discussion of Probability Relations between Separated Systems," *Math. Proc. Cambridge Philos. Soc.*, vol. 31, №. 4, P.P. 555–563, 1935, DOI: 10.1017/S0305004100013554.

[15] E. Schrödinger, "Probability relations between separated systems," *Math. Proc. Cambridge Philos. Soc.*, vol. 32, №. 3, P.P. 446–452, 1936, DOI: 10.1017/S0305004100019137.

[16] A. Einstein, B. Podolsky, and N. Rosen, "Can Quantum-Mechanical Description of Physical Reality Be Considered Complete?," *Phys. Rev.*, vol. 47, №. 10, P.P. 777–780, May 1935, DOI: 10.1103/PhysRev.47.777.

[17] A. K. Ekert, "Quantum Cryptography and Bell's Theorem," P.P. 413–418, 1992, DOI: 10.1007/978-1-4615-3386-3_34.

[18] J. L. Park, "The concept of transition in quantum mechanics," *Found. Phys.*, vol. 1, №. 1, P.P. 23–33, 1970, DOI: 10.1007/BF00708652.

[19] W. K. Wootters and W. H. Zurek, "A single quantum cannot be cloned," *Nature*, vol. 299, №. 5886, P.P. 802–803, Oct. 1982, DOI: 10.1038/299802a0.

[20] W. Heisenberg, "Über den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik," *Zeitschrift für Phys.*, vol. 43, №. 3–4, P.P. 172–198, 1927, DOI: 10.1007/BF01397280.

[21] C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution

and coin tossing," *Theor. Comput. Sci.*, vol. 560, №. P1, P.P. 7–11, 2014, DOI: 10.1016/j.tcs.2014.05.025.

[22] P. Sibson *et al.*, "Chip-based quantum key distribution," *Nat. Commun.*, vol. 8, 2017, DOI: 10.1038/ncomms13984.

[23] M. Lucamarini, Z. L. Yuan, J. F. Dynes, and A. J. Shields, "Overcoming the rate–distance limit of quantum key distribution without quantum repeaters," *Nature*, vol. 557, №. 7705, P.P. 400–403, May 2018, DOI: 10.1038/s41586-018-0066-6.

[24] Z. Yuan *et al.*, "10-Mb/s Quantum Key Distribution," *J. Light. Technol.*, vol. 36, №. 16, P.P. 3427–3433, 2018, DOI: 10.1109/JLT.2018.2843136.

[25] H.-K. Lo, M. Curty, and B. Qi, "Measurement-Device-Independent Quantum Key Distribution," *Phys. Rev. Lett.*, vol. 108, №. 13, p. 130503, Mar. 2012, DOI: 10.1103/PhysRevLett.108.130503.

[26] C. H. Park *et al.*, "Practical plug-and-play measurement-device-independent quantum key distribution with polarization division multiplexing," *IEEE Access*, vol. 6, P.P. 58587–58593, 2018, DOI: 10.1109/ACCESS.2018.2874028.

[27] B. K. Park, M. K. Woo, Y.-S. Kim, Y.-W. Cho, S. Moon, and S.-W. Han, "User-independent optical path length compensation scheme with sub-nanosecond timing resolution for a 1 × N quantum key distribution network system," *Photonics Res.*, vol. 8, №. 3, p. 296, Mar. 2020, DOI: 10.1364/PRJ.377101.

[28] T. A. Eriksson *et al.*, "Crosstalk Impact on Continuous Variable Quantum Key Distribution in Multicore Fiber Transmission," *IEEE Photonics Technol. Lett.*, vol. 31, №. 6, P.P. 467–470, 2019, DOI: 10.1109/LPT.2019.2898458.

[29] Власов Е.Г., *Конечные поля в телекоммуникационных приложениях. Теория и применение FEC, CRC,M-последовательностей*. Москва, 2016.

[30] *Theory and Practice of Error Control Codes*. Addison–Wesley Press, 1983.

[31] R. W. Hamming, "Error Detecting and Error Correcting Codes," *Bell Syst. Tech. J.*, vol. 29, №. 2, P.P. 147–160, Apr. 1950, DOI: 10.1002/j.1538-7305.1950.tb00463.x.

[32] G. Brassard and L. Salvail, “Secret-key reconciliation by public discussion,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 765 LNCS, P.P. 410–423, 1994, DOI: 10.1007/3-540-48285-7_35.

[33] W. T. Buttler, S. K. Lamoreaux, J. R. Torgerson, G. H. Nickel, C. H. Donahue, and C. G. Peterson, “Fast, efficient error reconciliation for quantum cryptography,” *Phys. Rev. A - At. Mol. Opt. Phys.*, vol. 67, №. 5, p. 8, 2003, DOI: 10.1103/PhysRevA.67.052303.

[34] R. Gallager, “Low-density parity-check codes,” *IEEE Trans. Inf. Theory*, vol. 8, №. 1, P.P. 21–28, Jan. 1962, DOI: 10.1109/TIT.1962.1057683.

[35] D. J. C. Mackay and R. M. Neal, “Good codes based on very sparse matrices,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 1025, P.P. 100–111, 1995, DOI: 10.1007/3-540-60693-9_13.

[36] D. J. C. MacKay, “Good error-correcting codes based on very sparse matrices,” *IEEE Trans. Inf. Theory*, vol. 45, №. 2, P.P. 399–431, Mar. 1999, DOI: 10.1109/18.748992.

[37] J. Martinez-Mateo, D. Elkouss, and V. Martin, “Key reconciliation for high performance quantum key distribution,” *Sci. Rep.*, vol. 3, P.P. 3–8, 2013, DOI: 10.1038/srep01576.

[38] N. Walenta *et al.*, “A fast and versatile quantum key distribution system with hardware key distillation and wavelength multiplexing,” *New J. Phys.*, vol. 16, №. 1, p. 013047, Jan. 2014, DOI: 10.1088/1367-2630/16/1/013047.

[39] M. Milicevic, C. Feng, L. M. Zhang, and P. G. Gulak, “Quasi-cyclic multi-edge LDPC codes for long-distance quantum cryptography,” *npj Quantum Inf.*, vol. 4, №. 1, P.P. 1–9, 2018, DOI: 10.1038/s41534-018-0070-6.

[40] I. B. Djordjevic, *Physical-Layer Security and Quantum Key Distribution*. 2019.

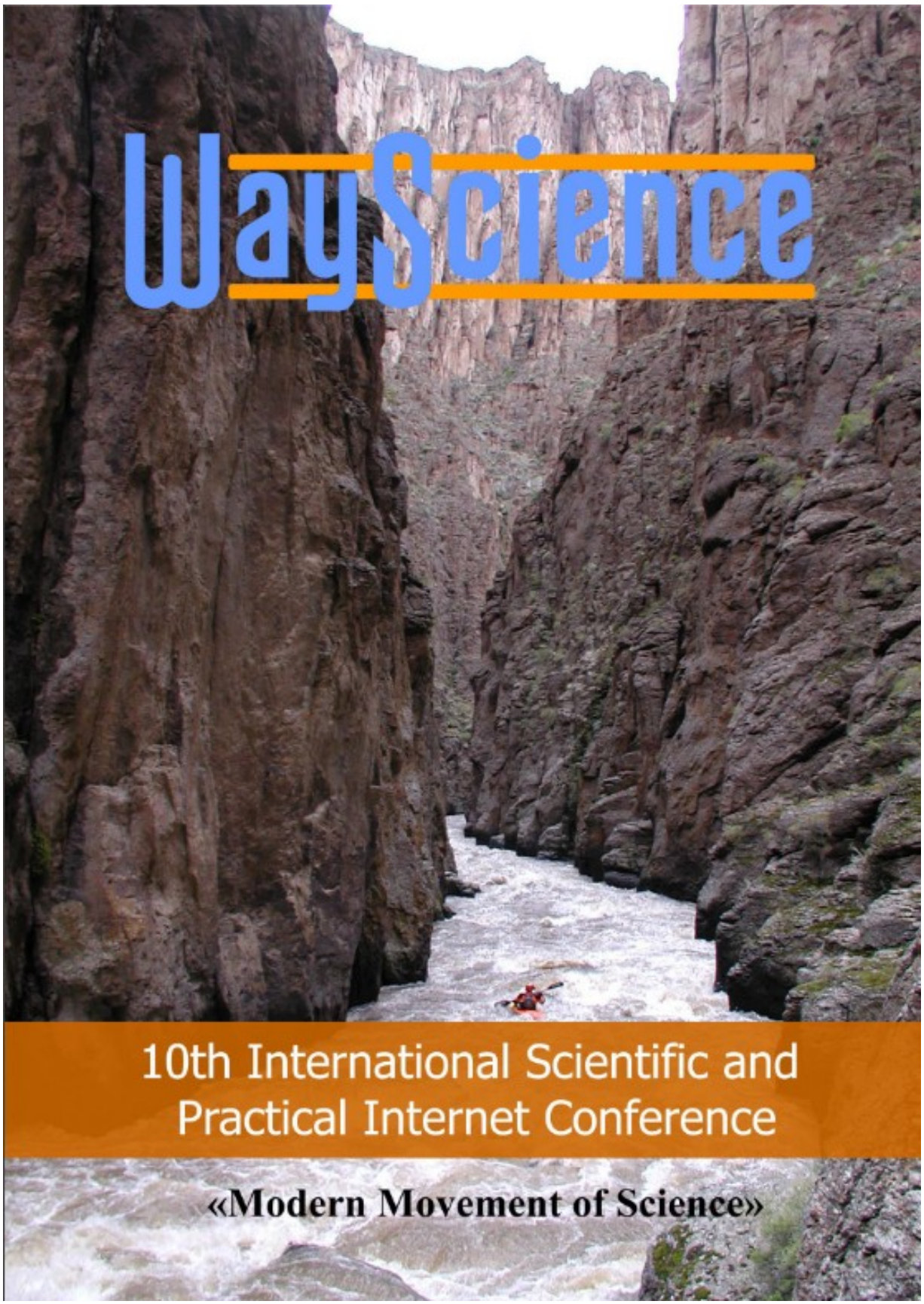
[41] M. Baldi, *QC-LDPC Code-Based Cryptography*. .

[42] Radford M. Neal, “Software for Low Density Parity Check Codes

[Електронний ресурс] - режим доступу:
www.cs.toronto.edu/~radford/ftp/LDPC-2012-02-11/index.html,” 2012.
[Online]. Available:
www.cs.toronto.edu/~radford/ftp/LDPC-2012-02-11/index.html%0D.

Додаток А


Міжнародний електронний
науково-практичний журнал "WayScience"
Дата проведення:
2-3 квітня 2020 року
СЕРТИФІКАТ
учасника конференції
X Міжнародна науково-практична інтернет-конференція
«СУЧАСНИЙ РУХ НАУКИ»
учасник
Білаш Богдан
Тема: «THE IMPLEMENTATION OF THE MODIFIED ERROR CORRECTION METHOD IN QUANTUM KEY DISTRIBUTION»
м. Дніпро (Україна) – 2020 р
Редакція журналу


WayScience
10th International Scientific and Practical Internet Conference
«Modern Movement of Science»

THE IMPLEMENTATION OF THE MODIFIED ERROR CORRECTION METHOD IN QUANTUM KEY DISTRIBUTION

Bilash Bohdan

National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", student, bogdanbelash35@gmail.com

1. Purpose and tasks

Purpose: research of error correction methods in quantum key distribution (QKD) systems. QKD is a system that uses photons and the laws of quantum physics to create the encryption key used in modern cryptography [1]. Unlike modern classical cryptographic protocols, QKD protocol is based on the creation of truly random keys that can be used in one-time pads that have the property of absolute cryptographic stability [2].

Tasks: to research an error correction method using one-time pad system to create fresh key.

2. Object and subject of research

Object: discrete-variable QKD (DV-QKD) system based on single photon polarization.

Subject: low density parity check (LDPC) codes to correct errors in sifted key.

3. Research methods and tools

Firstly, program code on C language is created. In the future, the method will be research between real Alice and Bob systems.

4. The algorithm of implementation of the modified method.

Method is based on [3]. Mario Milicevic proposed to use continuous-variable QKD (CV-QKD), where Alice encodes her information in the amplitude and phase quadratures of coherent states. Then, classic message M is created and sent to Alice by modulating code word C with a correlated Gaussian sequence Y.

Representations of vectors X and Y in the form of correlated Gaussian sequences for DV-QKD are not suitable, because these vectors must be discrete. After exchanging photons and reconciling the photon lengths and positions, Alice and Bob change them into the classic bits of "0" and "1", which does not to save fresh keys as correlated of Gaussian sequences.

Consider Fig. 1. It is a diagram of the algorithm of implementation of the modified method. Traditionally, the part that sends the message is called Alice, and the part that receives the message is called Bob. Alice and Bob have randomly generated X_0 and Y_0 keys of a certain length. It is proposed to reduce the number of bits to a certain fixed length *N*. These will be the new vectors X and Y.

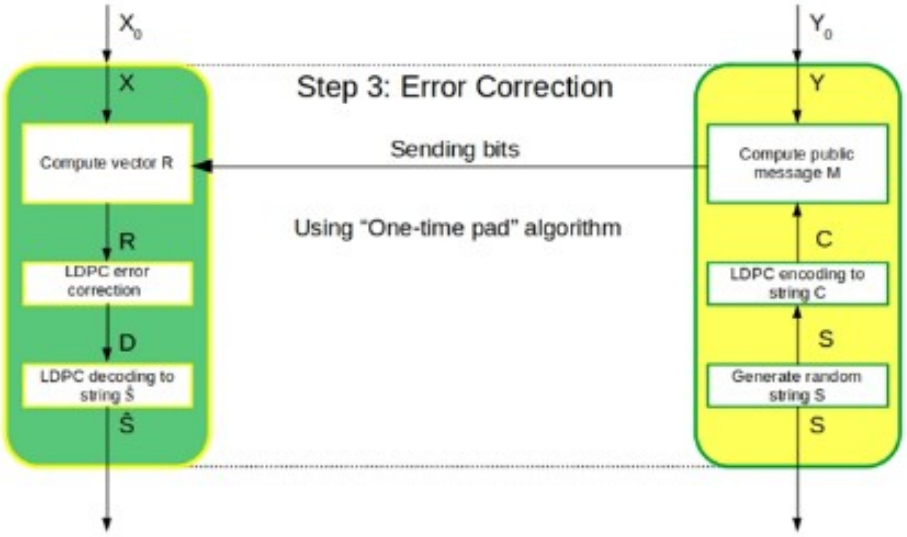


Figure 1. Diagram of the algorithm of implementation of the modified method.

Let's take a closer look at the proposed algorithm step-by-step:

1. Bob, using a random number generator, generates a message S of length $N/2$.

2. Using LDPC, Bob encodes the message S into a code word C. The last half of the bits of the code word are S.

3. Now, using a logical XOR operation between vectors C and Y, Bob prepares an M message to send it to Alice.

4. Bob sends an M message to Alice, who receives it.

5. Alice uses the XOR operation with vectors M and X to obtain the vector R.

Since the XOR operation is the same in both directions, the vectors R and C should be exactly the same, but they will be different in the positions where the vectors X and Y are different (the classical authentic channel for transmitting information is absolutely reliable). The example below shows that the positions where X and Y differ are also C and R.

6. Alice applies LDPC error correction to obtain vector D. After error correction, this vector is absolutely the same as the C code word.

7. Alice counts the number of bits that are different between vectors R and D. She then divides the number of bits by the total number of bits N. The value obtained is the error rate. If its value does not exceed 11%, then we can assume that Eve does not listen to us. This stage is the most important.

8. Alice decodes vector D and finds a message $\hat{S}$ that is exactly equal to Bob's original message S. These vectors are transmitted to the next stage to privacy amplification.

6. Results and conclusions.

Developed program is written by C language and uses the modified method of creating matrices suggested by the Radford Neal [4, 5]. Then, unlike [3], program creates the one-time pad [6] by using simple logical operation XOR to encode the message. The following is noted: this code needs a really random key. QKD allows to obtain such a key. If Eve does not listen to Alice and Bob, then this key is also known only to Alice and Bob. The solution to use Alice's and Bob's shifted keys is not for their reconciliation and subsequent use at the privacy amplification stage, but as a key in the one-time fall to reconcile the message, which will continue to be used for privacy amplification is proposed.

References:

1. C. H. Bennett, G. Brassard, "BB84highest.pdf," // *Proceedings of IEEE International Conference on Computers, Systems and Signal Processing*. P.-174–179, 1984.

2. C. E. Shannon, "Communication Theory of Secrecy Systems," *Bell Syst. Tech. J.*, vol. 28, №.4, .P.656–715,1949, DOI:10.1002/j.1538-7305.1949.tb00928.x.

3. M. Milicevic, C. Feng, L. M. Zhang, and P. G. Gulak, "Key Reconciliation with Low-Density Parity-Check Codes for Long-Distance Quantum Cryptography" April, P.P. 1–23, 2017, DOI: 10.1038/s41534-018-0070-6.

4. D. J. C. Mackay and R. M. Neal, "Good codes based on very sparse matrices," // in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1995, vol. 1025, P.P. 100–111, DOI: 10.1007/3-540-60693-9_13.

5. Radford M. Neal, "Software for Low Density Parity Check Codes." 2012.

6. W. Peng, D. Cheng, and C. Song, "One-time-pad cryptography scheme based on a three-dimensional DNA self-assembly pyramid structure," *PLoS One*, Vol. 13, №. 11, P. e0206612, Nov. 2018, DOI: 10.1371/journal.pone.0206612.

Додаток Б

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського
науково-технічний журнал «Мікросистеми, Електроніка та Акустика»
вул. Політехнічна 16 корп. 12, кім. 116, Київ, Україна, 03056

Довідка

Довідка видана Белаш Богдану про те, що його стаття «Optimal low density parity check matrices to correct quantum key errors for QKD» подана в науково-технічний журнал «Мікросистеми, Електроніка та Акустика» з реєстраційним номером 201253 та на поточний час проходить процедуру наукового рецензування.

14 травня 2020 р.

Відповідальний редактор
к.т.н. доц. **Олексій В. БОГДАНОВ**

Optimal low density parity check matrices to correct quantum key errors for QKD

Bilash Bohdan[1,2].

1. National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

2. Korea Institute of Science and Technology.

bogdanbelash35@gmail.com.

Abstract — In this paper, the parity-check matrices that can be used in quantum key distribution systems are analyzed. The quantum key distribution system has inevitable noises that must be corrected by an error correction algorithm to create a secure key. 1000 bits of sifted key are fed into the error correction algorithm at once, which is divided into 50 parts. The algorithm creates 50 syndromes corresponding to each part by multiplying 10 × 20 bit parity-check matrices. The algorithm sends the generated syndrome to the other side, which also divides the sifted key into 50 parts, creates a syndrome from each part, and compares with the received syndrome. If the syndromes are different, these sifted key parts are discarded. However, there may be situations where different parts may have the same syndromes. Therefore, it is necessary to find such an optimal matrix that the probability of getting the same syndromes at different parts of the sifted key is minimal.

Key words — QKD; LDPC; error correction; parity-check matrix; post-processing

1.1. Quantum key distribution.

Quantum key distribution (QKD) is a system that can securely share an identical key between two distant parties, Alice and Bob, which is applicable to

modern cryptography [1]. Unlike modern classical cryptographic protocols, such as RSA (Rivest–Shamir–Adleman), which is based on the practical difficulty of the factorization of the product of two large prime numbers [2], QKD protocol is based on the creation of truly random keys.

Although the first BB84 protocol [3] for QKD was proposed in 1984, this trend is new in modern science and is being actively researched. There are some subjects for research works on the QKD such as chip-scale system [4], long-distance communication [5], high secure key rate QKD [6]-[8], and efficient post-processing [9].

The main task of the post-processing is error correction to share an identical secure key with Alice and Bob. Therefore, it is necessary to explain where the errors occur. The unreliable quantum channel is named because photons may cause noise to change the photon polarization vector. There may also be errors when accepting photons by Bob and misreading the state of the photon. The single-photon detecter, which is an extremely sensitive component for detecting a single photon, has inevitably has some noises, such as Dark count, After pulse, also there is Cross talk from the other channel [10]. Theoretically, the secure key rate is almost zero when the quantum bit error rate (QBER) is more than 11%. Usually, the QBER from average background noises of the system is under 5% which must be corrected. That is why they use error correction at the post-processing stage. It should also be noted that at the post-processing stage and at subsequent stages, the exchange of information between Alice and Bob takes place via a classical channel, which with a high degree of probability can be considered as 99.99999% of reliability. Therefore, the task is to correct the errors that occurred during the phase of the photon exchange.

1.2. Analysis of existing error correction methods.

There are some error correction methods to correct quantum bit error, such as Cascade, Winnow, Low-Density Parity-Check codes (LDPC). Those ideas are adopted from the classical error correction methods.

In the Cascade [11] protocol, in each pass, Alice and Bob agree on a random permutation that applies to their bits.

Winnow [12], like Cascade, breaks binary strings to match them into blocks, but instead of bug fixing using iterative binary bug fixing is based on Hamming code.

But these protocols work poorly over long distances, also with long-distance messages. It is necessary to use a protocol that would contain the check bits together with the main message at one time during transportation. The countermeasure of those problems is the LDPC protocol [13]. Nowadays, the computing power and electronics have improved, so the LDPC protocol has a lot of attention and developments on the LDPC code [14], [15].

Figure 1 is the overall procedure of the QKD. At the error correction step, Alice and Bob have sifted keys, but they are slightly different from each other due to the background noises of the QKD system. The purpose of this phase is to reconcile the sifted keys so that they are the same and then pass them on to the next stage of privacy amplification. The main problem at this stage is that when transferring the sifted keys between Alice and Bob, it is necessary to protect them so that Eve could not get the sifted key. To avoid this, you can use hashing protocols to protect the information being transmitted. But this approach is not rational because of increasing computing resources and the processing time for the hash function. That is why there is another approach for correcting errors and protecting information from Eve.

Consider the known methods for error correction. In [3], researchers propose a full-cycle creating QKD system. According to their error-correction step, one side of the fresh key encodes a syndrome and sends it to the other side. The other side decodes the syndrome and compares the results with its own. If results do not match, this key is discarded and a new one is sent. We assume under certain conditions even fresh keys are different, they can have the same syndromes. In this work, we will analyze more detailed cases to do not have the same syndromes.

2.1. Common syndromes.

In the classical LDPC applying, first, the parity-check matrix H is generated, in which the “1” are uniformly and very rarely located, and all other positions are “0”. Such a matrix is not systematic. By using Gaussian elimination it is necessary to transform it into a systematic form $[-P^T | I_{n-k}]$, where I_{n-k} is a prime identity matrix, P^T — binary matrix (in binary codes -P = P). From matrix H we creating generate matrix $G = [I_k | P]$. Usually, one side, Alice, has an output message m of size k that is converted into a codeword c of length n by a matrix G. Therefore, the matrix G has dimensions $k \times n$. The k/n ratio is called the relative code transmission rate (or just code rate) [4]. Typically, this rate is 1/2, 2/3, 3/4 and so on. The codeword is sent to the other side, Bob, through the noise channel. Bob accepts a vector r, which may differ from the codeword by some number of bits. In a simple example, the resulting vector differs by one bit. Bob, using the matrix multiplication of matrix H and the resulting vector r, obtains a vector s called a syndrome: s = H*r. If, as a result, the syndrome has all zeros, then the resulting vector r has no errors and is equal to the codeword. Bob then decodes the vector into a message m that is equal to Alice's message. If the vector r, in the simplest case, has one error, then the syndrome s will coincide with the column of the matrix H, whose number will be the number of the error bit in the vector r.

In QKD systems, LDPC codes cannot be used as they would in classic applications. You cannot just send a sifted key because eavesdropper (Eve) can find out about the sifted key and then there is no point in creating a secure key. Therefore, it is necessary to come up with such a method of correcting the errors in Alice and Bob's sifted keys so that Eve cannot find out about them. In [3] it is suggested to create and exchange syndromes between the sides. This idea needs to be explained in more detail.

In Figure 1, in the first step, Alice encodes her information in the polarization of single-photon states and sends it to Bob. Bob detects single-photon states with the selected base and records the measurement result into classical bits. After that, Bob sends the chosen bases to Alice, who in turn discards the classical bits whose bases

did not match with the chosen Bob [5]. As a result, at the beginning of step 3, Alice and Bob have sifted keys of the same length, but which differ by a certain percentage of bits. This is called the quantum bit error rate (QBER).

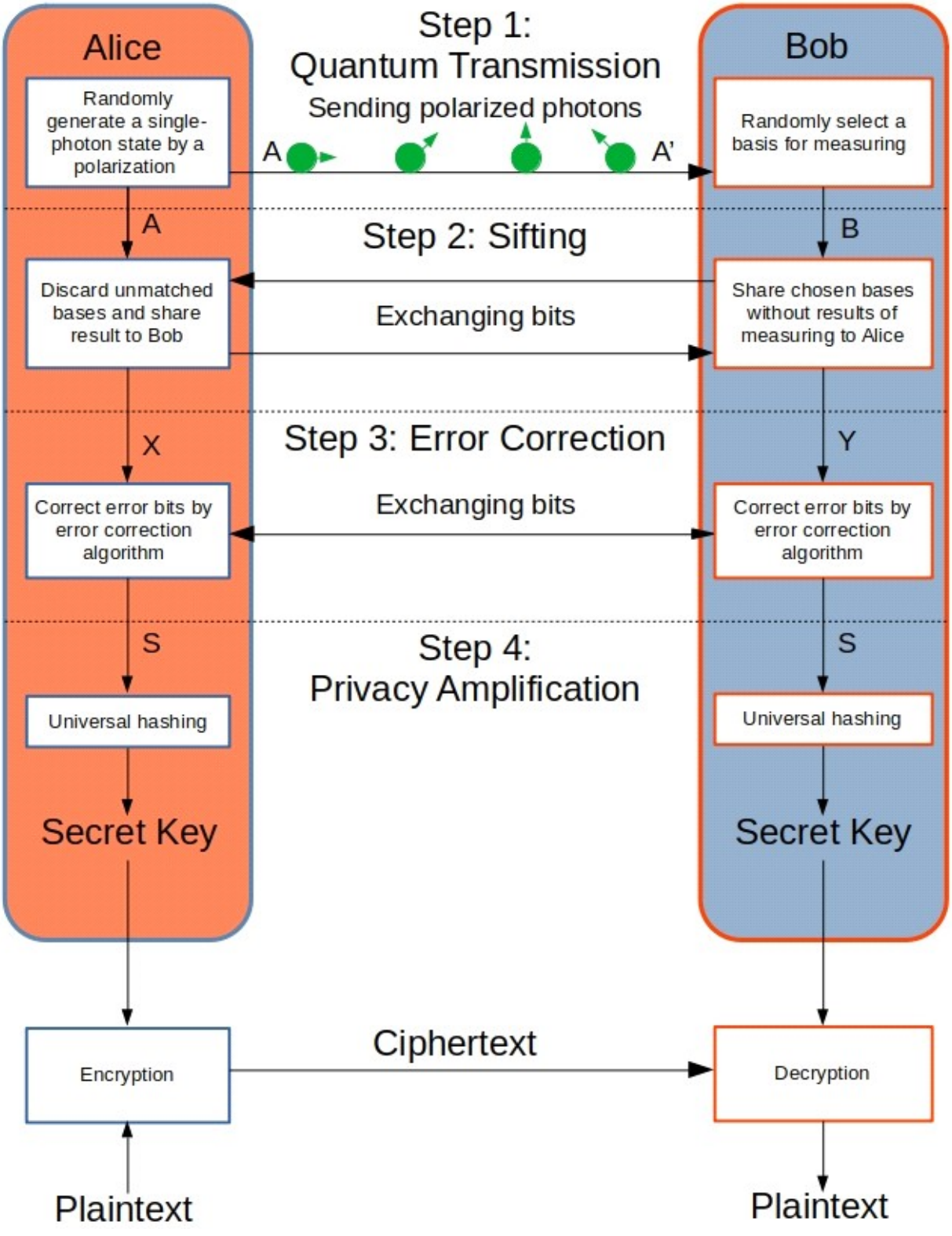


Figure 1. Overall procedure for the QKD.

Drawing on the analogy of the classic error correction method, QBER is additive white Gaussian noise (AWGN).

In the case of QKD, the two sides already have sifted keys, X and Y, which are not the same and need to be fixed. Alice cannot send her sifted key to Bob, because Eve immediately finds out about it. Therefore, it is suggested to exchange syndromes. A syndrome is created by multiplying the sifted key by matrix H: $s = HX^T$. The proposed method does not require the creation of a matrix G. Unlike the classic case, the syndrome is unknown in advance (but can also be all zeros). It is not necessary to correct the sifted key X with the resulting s syndrome. This syndrome is sent to Bob. Eve, having received the syndrome has nothing to do with it. Bob creates his syndrome $\hat{s}$ by multiplying his sifted key by the matrix H: $\hat{s} = HY^T$. The main idea is that if parts of the sifted keys are the same - the created syndromes will also be the same. If parts of the sifted keys are different, the syndromes should be different and these parts of sifted keys will be discarded.

We used a 10 × 20 parity check matrix. This is the standard matrix size at which the coding rate in the classical method is 1/2 and is one of common. In this case, the length of the sifted key is 20 bits, and the created syndrome is 10 bits. Then the possible combinations of the sifted key may be 2^{20}, and the possible combinations of syndrome 2^{10}. In this case, one syndrome will respond to 1024 different messages. Therefore, when the messages are different, there may be situations where the syndromes will be the same. Then the error correction of the part will be failed. The solution may be to increase the length of the syndrome or to find a special H matrix, which, under certain conditions, will not cause common syndromes for different messages.

2.2. Shared messages and their dependency on QBER.

To analyze the error correction algorithm, we arbitrarily selected messages of

1000 bits length. After that, we randomly changed the message bits depending on QBER. The simulation program was written in C language. The algorithm for adding error bits, for example, for QBER = 5% is as follows: 1) we generate a random number from 0 to 999 for each bit of the sifted key (1000 bits total), 2) if this number is less than 50 (in our case it is equivalent to 5% for QBER) - change this bit; if more - we save this bit. This does not mean that it will change exactly 50 bits in the message, but in Figure 2 it will be similar to a Poisson distribution with a mathematical expectation of 50 bits. For each QBER from 0.1% to 25% in 0.1% increments (1 to 250 bits, 1-bit increments), we repeated this procedure 100 times.

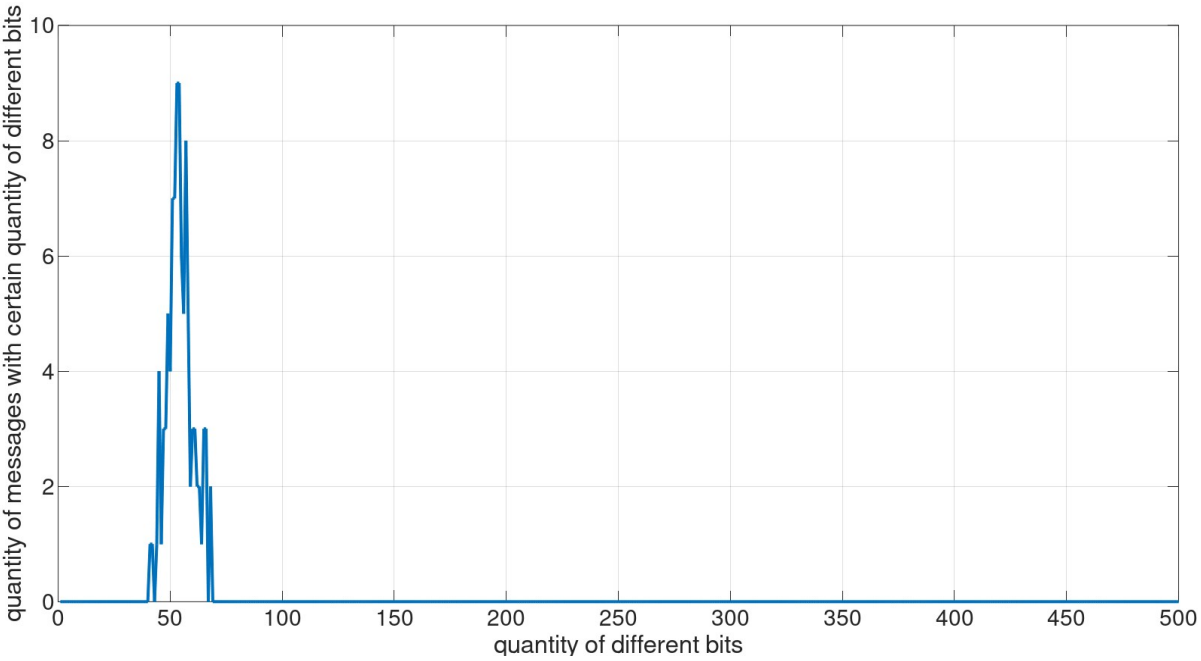


Figure 2. Changed bit distribution for 5% QBER.

As a result, we have two messages: the original message, which is the sifted key of Alice, X, and changed to a certain number of bits equal to QBER, the message that is the sifted key of Bob, Y.

Next, we divide each sifted key from Alice and Bob into 50 parts, 20 bits each, and compare these parts. Then count the number of parts that are the same. Ideally, there will be 50 (all parts of one sifted key are equal to all parts of another sifted key).

Figure 3 shows the average number of messages that will be the same for

different QBER. According to the graph, one can see that at a very small value of QBER most parts of the sifted key are the same. At QBER = 5%, almost 2/3 of the messages will be discarded.

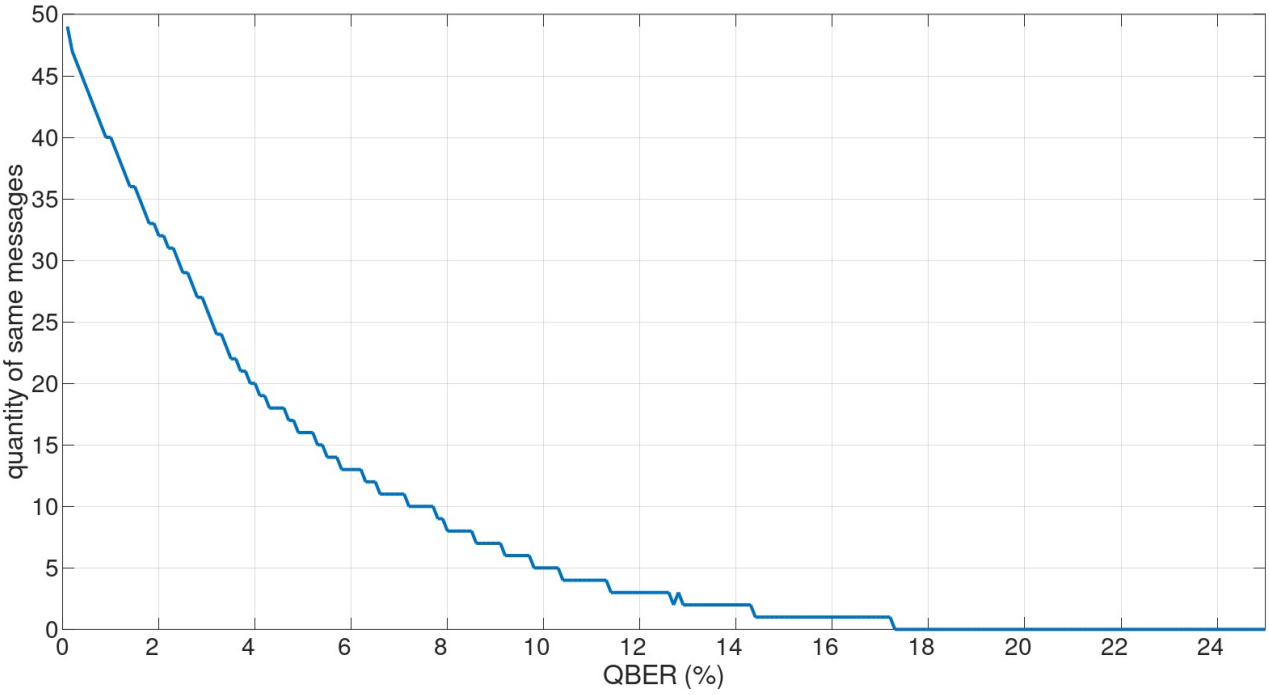


Figure 3. Identical parts of the message depending on QBER.

2.3. Investigation of conflicting messages.

As stated above, even if parts of sifted keys are different, there may be situations where syndromes are the same.

At QBER = 5%, we expect our sifted keys to be 50 bits apart. Of course, there may be a situation where these 50 bits are stocked at three parts, and then each part will contain at least 10 changed bits. But a more realistic situation is when these bits are distributed evenly between all parts. Then one bit will be changed in each part.

We created a new syndrome from the part of the sifted key and the matrix H, which is the same for all parts of the message. The matrix was created using the method suggested by David Mackay and Radford Neal in [1], [2]. Radford Neal has created open-source software [18]. We have integrated Neal's code to generate the H matrix into our code.

Since the created H matrix has dimensions of 10 × 20 bits, it is enough for checking all combinations of 20-bit messages. Therefore, the need to investigate all combinations of errors of the original key of 1000 bits in length makes no sense. It is enough to investigate all the combinations of a part of the message that has only 20 bits.

Since the H matrix must be sparse, the number of "1" in the matrix should at least not exceed the number of "0". Therefore, we have created 10 matrices using Neal's software, 5 matrices for "evencol" and 5 matrices for "evenboth" methods. In each matrix, the number of "1" in each column is from 1 to 5. The seed value of the random function is 1.

We used a message in which all 20 bits are zeros and changed the bits in every possible combination. First, we changed one bit in each possible position, then two, three, etc., to consider all possible combinations of the modified message. They then created the syndromes from these messages and compared them with the original message's syndrome. We calculated the number of common syndromes varying a certain number of bits. The results showed that with the "evenboth" method and number of 1 in each column is 4, which is optimal H matrix for applying the QKD system because QKD operates properly QBER is under 5%. The X-axis and Y-axis are changed bits from 0 to 20 and the number of common syndromes from the original and changed message, respectively.

According to the results, when the number of changed bits is less than 4, we do not have common syndromes in different messages. As mentioned above, errors are distributed in the sifted keys evenly, and therefore at QBER = 5%, the higher the probability that 1-3 bits will be changed in each message, and the lesser probabilities that more than 4 bits will be changed (Figure 4).

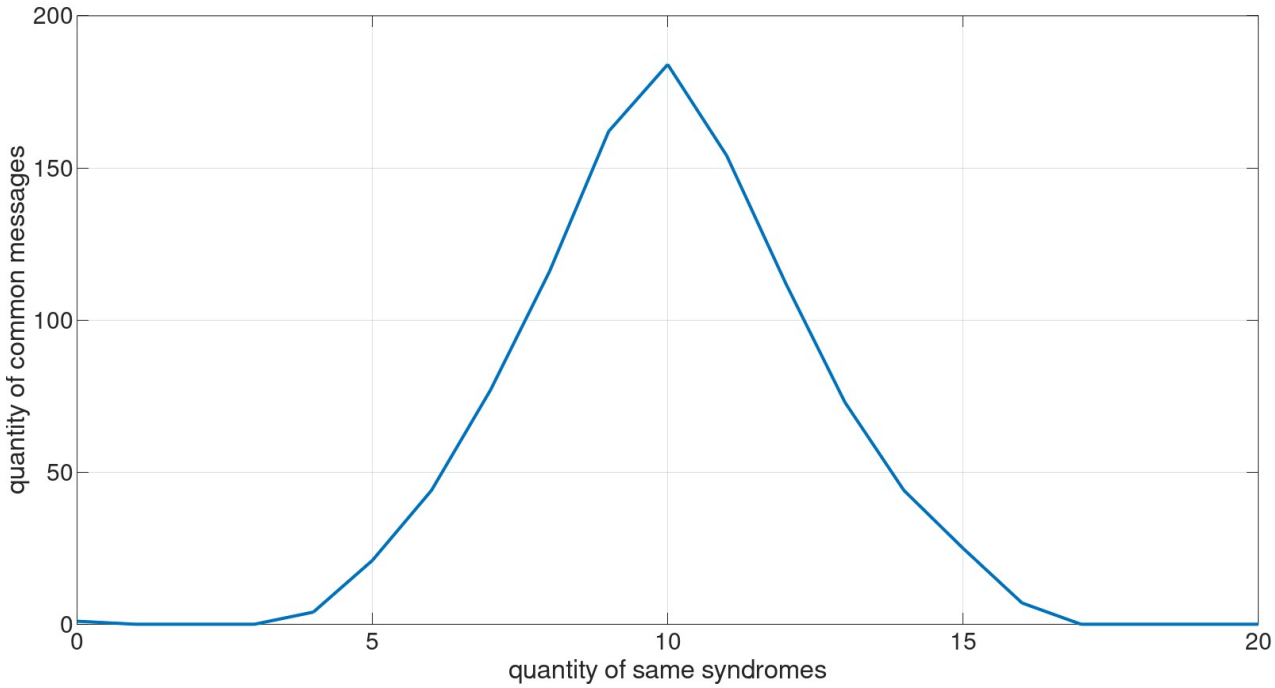


Figure 4. Distribution of the number of same syndromes versus the number of changed bits.

2.4. Seed research for Neal's matrices.

These results, as mentioned above, were obtained when the seed value for the generation of the matrix was 1. We conducted studies and created 1000 different matrices where the seed value was from 1 to 1000. Of these, 144 matrices did not have common syndromes, if they were only changed up to three bits in each message. So, theoretically, we can use those matrices to create syndromes to apply the QKD system.

Among these syndromes, when changing 4 bits into messages, there may be a common number of syndromes. For seed = 1, the number of common syndromes is 4. The total result for the 144 matrices found is in Figure 5.

As you can see from the graph, the total number of common syndromes is 4-6, but there are matrices when the number of common syndromes is 1 which is enough for QKD condition.

3. Results.

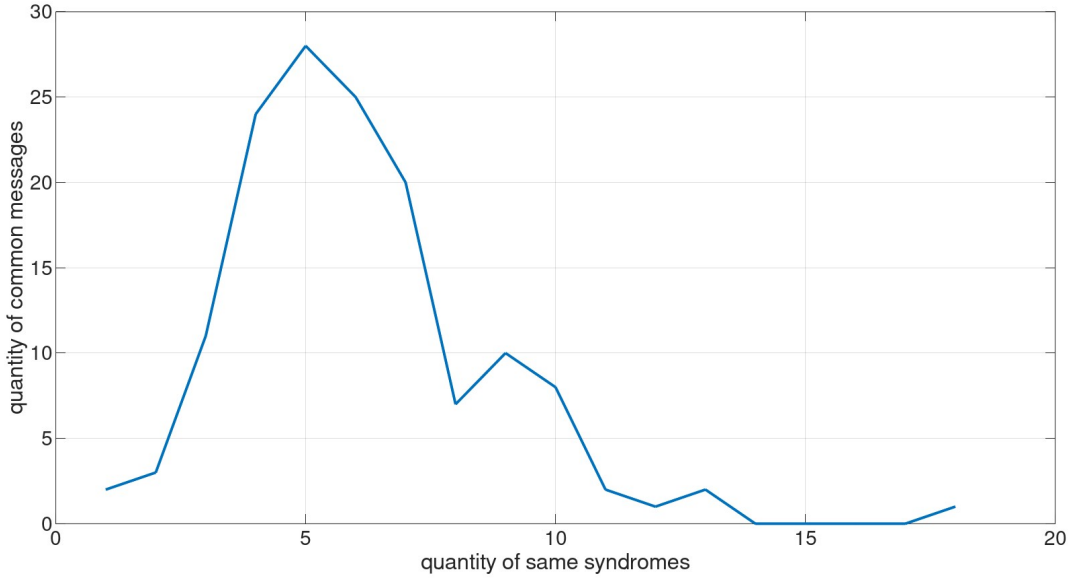
The complete error correction process is depicted in Figure 6.

4. Conclusion and discussion.

The main purpose was to investigate the error correction step proposed in [12]. As can be seen from Figure 3, this method is not rational, but under certain conditions, it is easier to reject most of the messages than to correct them. It is not possible to use a message length equal to the length of the message because Eve can easily find out about the message by converting the syndrome back into a message. Therefore, the classic model is chosen when the length of the syndrome is twice less than the length of the message. If the length of the message is 20 bits and the length of the syndrome is 10 bits, then 1024 different messages will be needed for one syndrome, so it has robustness against Eve's attack. In addition, such messages in our case are 50 pieces, so the probability of finding the right message increases exponentially. But in this case, common syndromes may occur with different reports. You can increase the number of bits in a syndrome, or find a matrix in which the number of common syndromes under certain. The optimal matrix for the QKD system generates different syndromes under three erroneous bits The generated 10 × 20 bit Neal's matrix, with 4 of "1" in each column, showed the best result in creating message syndromes to share an identical sifted key between Alice and Bob.

The process of discarding messages is not efficient enough, therefore the object of future research is to develop an algorithm for correcting errors in parts of the sifted key in which Alice's and Bob's syndromes are not matched.

Figure 5.



Number of common syndromes at 4 changed bits.

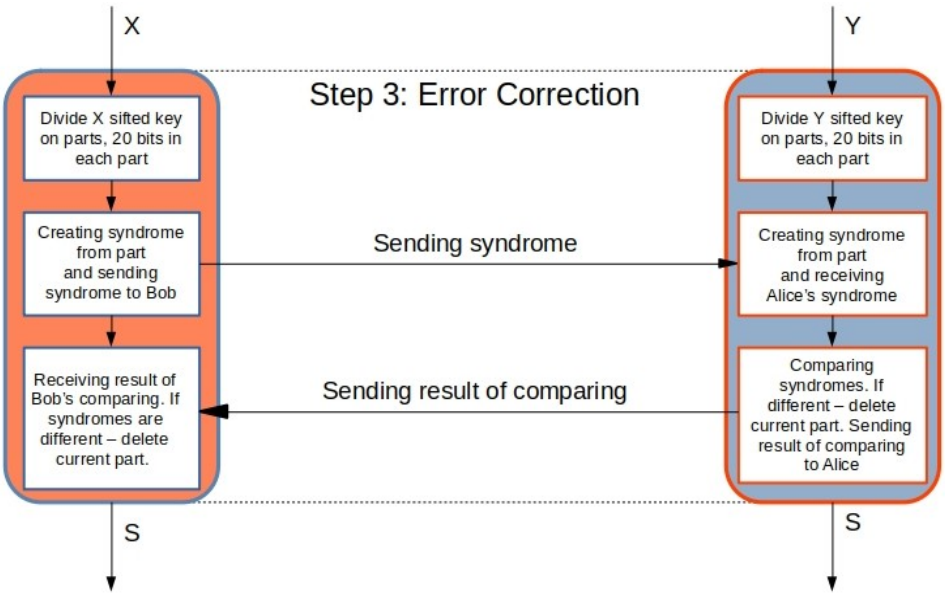


Figure 6. Error correction step.

References.

[2] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Commun. ACM, vol. 21, №. 2, P.P. 120–126, Feb. 1978, DOI: 10.1145/359340.359342.

[3] C. H. Bennett and G. Brassard, "BB84highest.pdf," Proceedings of IEEE International Conference on Computers, Systems and Signal Processing. P.P. 174–179, 1984.

[4] P. Sibson et al., "Chip-based quantum key distribution," Nat. Commun., vol. 8, №. May 2016, 2017, DOI: 10.1038/ncomms13984.

[5] M. Lucamarini, Z. L. Yuan, J. F. Dynes, and A. J. Shields, "Overcoming the rate–distance limit of quantum key distribution without quantum repeaters," Nature, vol. 557, №. 7705, P.P. 400–403, May 2018, DOI: 10.1038/s41586-018-0066-6.

[6] Z. Yuan et al., "10-Mb/s Quantum Key Distribution," J. Light. Technol., vol. 36, №. 16, P.P. 3427–3433, 2018, DOI: 10.1109/JLT.2018.2843136.

[7] H.-K. Lo, M. Curty, and B. Qi, "Measurement-Device-Independent Quantum Key Distribution," Phys. Rev. Lett., vol. 108, №. 13, p. 130503, Mar. 2012,

DOI: 10.1103/PhysRevLett.108.130503.

[8] C. H. Park et al., “Practical plug-and-play measurement-device-independent quantum key distribution with polarization division multiplexing,” IEEE Access, vol. 6, P.P. 58587–58593, 2018, DOI: 10.1109/ACCESS.2018.2874028.

[9] B. K. Park, M. K. Woo, Y.-S. Kim, Y.-W. Cho, S. Moon, and S.-W. Han, “User-independent optical path length compensation scheme with sub-nanosecond timing resolution for a 1 × N quantum key distribution network system,” Photonics Res., vol. 8, №. 3, p. 296, Mar. 2020, DOI: 10.1364/PRJ.377101.

[10] T. A. Eriksson et al., “Crosstalk Impact on Continuous Variable Quantum Key Distribution in Multicore Fiber Transmission,” IEEE Photonics Technol. Lett., vol. 31, №. 6, P.P. 467–470, 2019, DOI: 10.1109/LPT.2019.2898458.

[11] G. Brassard and L. Salvail, “Secret-key reconciliation by public discussion,” Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 765 LNCS, P.P. 410–423, 1994, DOI: 10.1007/3-540-48285-7_35.

[12] W. T. Buttler, S. K. Lamoreaux, J. R. Torgerson, G. H. Nickel, C. H. Donahue, and C. G. Peterson, “Fast, efficient error reconciliation for quantum cryptography,” Phys. Rev. A - At. Mol. Opt. Phys., vol. 67, №. 5, p. 8, 2003, DOI: 10.1103/PhysRevA.67.052303.

[13] Gallager, “Low density parity check codes,” 1963.

[14] D. J. C. Mackay and R. M. Neal, “Good codes based on very sparse matrices,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 1995, vol. 1025, P.P. 100–111, DOI: 10.1007/3-540-60693-9_13.

[15] D. J. C. MacKay, “Good error-correcting codes based on very sparse matrices,” IEEE Trans. Inf. Theory, vol. 45, №. 2, P.P. 399–431, 1999, DOI: 10.1109/18.748992.

[16] N. Walenta et al., “A fast and versatile quantum key distribution system with hardware key distillation and wavelength multiplexing,” New J. Phys., vol. 16,

2014, DOI: 10.1088/1367-2630/16/1/013047.

[17] W. C. Huffman and V. Pless, Fundamentals of Error-Correcting Codes. Cambridge University Press, 2003.

[18] Radford M. Neal, “Software for Low Density Parity Check Codes.” 2012.

Додаток В

[여기에 입력]

KIST Korea Institute of Science and Technology

Center for Quantum Information

Review on master thesis investigation
topic: Method of the LDPC-based key reconciliation in a quantum key distribution system

student: BILASH Bohdan Olegovych

Bohdan Bilash has researched quantum key distribution from March 2019 to May 2020 in the center for quantum information, KIST. He has written a document summarizing this work.

This work has started with a theoretical study to deepen his knowledge about an error correction using low-density parity-check codes in quantum key distribution systems that are used in this project. And the work includes an experimental part, based on creating a special parity check matrix and implementing it to error correcting algorithm to keep secure between legal parts. By means of this experimental approach, he has been able to compare different previously published techniques of low-density parity-check codes implementation. He has also figured out how to apply it to quantum key distribution system. Bohdan Bilash has then focused on the evolution error correction algorithm to create a more secure quantum key distribution protocol. Based on this study, he has been able to show that the more efficient error correction algorithm may be created. Finally, Bohdan Bilash proposed a better error correction matrix, that let us correct less than 3 errors in every part of the sifted key, which let to create a more efficient quantum key distribution system.

As a conclusion, I consider that Bohdan Bilash's work is highly valuable. Conducted with a rigorous scientific and engineering approach, this research will be helpful to design future devices in quantum communications. Taking all aspects into account, I consider the student deserves an "excellent" grade and assignment of a Master's degree.

Signature

Sang wook Han

Han, Sang-Wook
Director, Principal researcher, Ph. D
Center for Quantum Information
Korea Institute of Science and Technology (KIST)

Додаток Г.

```
/*
        Created by Bogdan Belash for KIST in 2020

*/

#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#include "program.h"

int main (void) {
        int i, j, d, r, c, k, m, h;
        int qber_c, message_c, bit_c, same_1, same_2, change_1, change_2, change_3,
found_m, good_c, good_c_2;
        int same_syndrome;
        int divided_my_S[50][20];                                   //array
with divided message from original 1000-bit message
        int changed_S [message];                                    //here  we
save all changed messages
        int divided_changed_s [50][20];
```

```
        int my_syndrome [10];                                   //variable
to save syndrome
        int changed_syndrome [10];                              //here  we
save all changed syndromes
        int new_s [20];
        int new_syndrome [10];
        int M = 10, N = 20;
        //quantity of rows and columns in H matrix
        char *file = "make-ldpc";                               //name  of
file, where save H matrix; inreadable for human
        long random_n;

        FILE *result_file, *summary_file, *summary_file_2;
        char result_name [100], summary_name [100], summary_name_2 [100];

        random_n = time(NULL);
        //initial seed from time

        //generate random fresh key from Alice
        gen_fresh_key ();

        //divide original message for 50 parts, 20 bits in every part
        d = 0;
        for (i = 0; i < 50; i++) {
```

```
        for (j = 0; j < 20; j++) {
                divided_my_S [i][j] = (int) my_S[d];
                d++;
        }
}

//creating 10x20 H matrix with 4 "ones" in each column, seed is 1
ldpc (file, M, N, my_seed, my_method, "4");
printldpc (file, 4);

sprintf (summary_name, "result.dat");
summary_file = fopen (summary_name, "w");
sprintf (summary_name_2, "result_2.dat");
summary_file_2 = fopen (summary_name_2, "w");

//creating error bits for original message
for (qber_c = 1; qber_c <= 250; qber_c++) {
//QBER from 0.1% to 25%
        fprintf (summary_file, "%d ", qber_c);
        fprintf (summary_file_2, "%d ", qber_c);

        good_c_2 = 0;
```

```
//100 repeats to check changed messages
for (message_c = 1; message_c <= 100; message_c++) {
        good_c = 0;

        sprintf (result_name, "results/%d/%d.dat", qber_c, message_c);
        result_file = fopen (result_name, "w");

        //creating message with error bits
        for (i = 0; i < message; i++) {
                srand(random_n);

                r = rand() % message;

                random_n = rand();
                //random_n has new number, because system time with 1 secound is not enough

                if (r < qber_c) {
                        if (0 == (int) my_S[i]) {
                                changed_S[i] = 1;
                        }
                        else {
                                changed_S[i] = 0;
```

```
                }


        }
        else {
                changed_S[i] = (int) my_S[i];
        }
}

same_syndrome = 0;

//divide changed message for 50 parts
d = 0;
for (i = 0; i < 50; i++) {
        for (j = 0; j < 20; j++) {
                divided_changed_s [i][j] = changed_S[d];
                d++;
        }

        //create syndrome from part of original message
        k = 0;
        m = 0;
        for (j = 0; j < 10; j++) {
                for (h = 0; h < 20; h++) {
```

```
                k = (int) my_H [j][h] * divided_my_S[i][h];
                m ^= k;
            }

            my_syndrome [j] = m;

            m = 0;
        }

        //create syndrome from part of changed message
        k = 0;
        m = 0;
        for (j = 0; j < 10; j++) {
            for (h = 0; h < 20; h++) {
                k = (int) my_H [j][h] * divided_changed_s [i]
[h];
                m ^= k;
            }

            changed_syndrome [j] = m;

            m = 0;
        }
```

```
                    //compare syndromes
                    c = 0;
                    for (j = 0; j < 10; j++) {
                            if (changed_syndrome [j] == my_syndrome [j]) {

                                    c++;
                            }
                    }

                    if (10 == c) {
//                          same_syndrome++;
                            fprintf (result_file, "In %d same SYNDROME\n", i +
1);

                            good_c++;
                    }
                    else {
                            found_m = 0;

                            for (change_1 = 0; change_1 < 20; change_1++) {
                                    for (j = 0; j < 20; j++) {
                                            new_s [j] = divided_changed_s [i][j];
                                    }
```

```
//changing bits in changed message to find original message

if (0 == new_s [change_1]) {
        new_s [change_1] = 1;
}
else {
        new_s [change_1] = 0;
}

//create new syndrome from original message
k = 0;
m = 0;
for (j = 0; j < 10; j++) {
        for (h = 0; h < 20; h++) {
                k = (int) my_H [j][h] * new_s [h];
                m ^= k;
        }

        new_syndrome [j] = m;

        m = 0;
}
```

```
                    same_1 = 0;

                    for (bit_c = 0; bit_c < 10; bit_c++) {
                        if (new_syndrome [bit_c] == my_syndrome [bit_c]) {
                            same_1++;
                        }
                    }

                    if (10 == same_1) {
                        same_2 = 0;

                        for (j = 0; j < 20; j++) {
                            if (new_s [j] == divided_my_S [i] [j]) {
                                same_2++;
                            }
                        }

                        if (20 == same_2) {
                            fprintf (result_file, "In %d ONE error\n", i + 1);
                            found_m = 1;
```

```
                    good_c++;
                    break;
                }
            }

            for (change_2 = change_1 + 1; change_2 < 20;
change_2++) {
                for (j = change_1 + 1; j < 20; j++) {
                    new_s [j] = divided_changed_s [i]
[j];
                }

                if (0 == new_s [change_2]) {
                    new_s [change_2] = 1;
                }
                else {
                    new_s [change_2] = 0;
                }

                k = 0;
                m = 0;
                for (j = 0; j < 10; j++) {
                    for (h = 0; h < 20; h++) {
```

```
                k = (int) my_H [j][h] *
new_s [h];
                m ^= k;
            }

            new_syndrome [j] = m;

            m = 0;
        }

        same_1 = 0;

        for (bit_c = 0; bit_c < 10; bit_c++) {
            if (new_syndrome [bit_c] ==
my_syndrome [bit_c]) {
                same_1++;
            }
        }

        if (10 == same_1) {
            same_2 = 0;

            for (j = 0; j < 20; j++) {
```

```
                    if (new_s [j] ==
divided_my_S [i][j]) {
                        same_2++;
                    }
                }

                if (20 == same_2) {
                    fprintf (result_file, "In %d
TWO errors\n", i + 1);
                    found_m = 1;
                    good_c++;
                    break;
                }
            }

            for (change_3 = change_2 + 1; change_3
< 20; change_3++) {
                for (j = change_2 + 1; j < 20; j++)
{
                    new_s [j] =
divided_changed_s [i][j];
                }

                if (0 == new_s [change_3]) {
```

```
            new_s [change_3] = 1;
        }
        else {
            new_s [change_3] = 0;
        }

        k = 0;
        m = 0;
        for (j = 0; j < 10; j++) {
            for (h = 0; h < 20; h++) {
                k = (int) my_H [j][h]
* new_s [h];
                m ^= k;
            }

            new_syndrome [j] = m;

            m = 0;
        }

        same_1 = 0;
```

```
                                        for (bit_c = 0; bit_c < 10; bit_c++)
{
                                                if (new_syndrome [bit_c] ==
my_syndrome [bit_c]) {
                                                        same_1++;
                                                }
                                        }

                                        if (10 == same_1) {
                                                same_2 = 0;

                                                for (j = 0; j < 20; j++) {
                                                        if    (new_s    [j]    ==
divided_my_S [i][j]) {
                                                                same_2++;
                                                        }
                                                }

                                                if (20 == same_2) {
                                                        fprintf (result_file, "In
%d THREE errors\n", i + 1);
                                                        found_m = 1;
                                                        good_c++;
                                                        break;
```

```
                                        }
                                    }
                                }
                            }
                        }

                        if (0 == found_m) {
                            fprintf (result_file, "In %d MORE errors\n", i +
1);
                        }
                    }
                }

                fclose (result_file);

                fprintf (summary_file, "%d ", good_c);

                good_c_2 += good_c;
            }

            fprintf (summary_file, "\n");
```

```
            fprintf (summary_file_2, "%d\n", good_c_2 / 100);
        }

        fclose (summary_file);
        fclose (summary_file_2);
}
```